

# DISENTANGLING NEURAL NETWORK REPRESENTATIONS FOR IMPROVED GENERALIZATION

A Thesis

Presented to

The Academic Faculty

By

Michael Cogswell

In Partial Fulfillment

Of the Requirements for the Degree

Doctor of Philosophy in Computer Science

Georgia Institute of Technology

*May 2020*

Copyright © 2020 by Michael Cogswell

# Disentangling Neural Network Representations for Improved Generalization

*Approved by:*

**Dr. Dhruv Batra**

School of Interactive Computing

Georgia Tech

**Dr. Devi Parikh**

School of Interactive Computing

Georgia Tech

**Dr. James Hays**

School of Interactive Computing

Georgia Tech

**Dr. Ashok Goel**

School of Interactive Computing

Georgia Tech

**Dr. Stefan Lee**

School of Electrical Engineering and Computer Science

Oregon State

Defense date: March 10, 2020

# Acknowledgements

A PhD is an academic journey and a personal journey. For better or worse, my grad school experience will shape my perspective on life and work as long as those things go on. Because of the people who helped me on my journey, I think it will be for the better.

First, thanks to Dhruv Batra and Devi Parikh who have worked as a team to support me and a league of other young, smart, motivated students. I was lucky that they turned out to be not just advisors, but also mentors. They taught me how to do research and funded me through my PhD. But they also showed me the importance of building trusting relationships with co-workers. And they showed me how to promote healthy disagreements. And they even helped me balance my work against my life despite my balance being different than their own.

Thanks to Stefan Lee, who also guided me both academically and personally. You helped me understand the world and my place in it.

Thanks to my peers from the CVMLP labs. To Jiasen Lu, who's seemingly endless motivation and talent can be infectious. To Rama Vedantam, for helping me learn to think and talk about things. And thanks to the rest of the CVMLP labs, especially including Yash Goyal, Aishwarya Agrawal, Stan Antol, Ar-

jun Chandrasekaran, Ashwin Kalyan, Abhishek Das, Ram Selvaraju, Nirbhay Modhe, and Erik Wijmans.

Thanks to Ashok Goel, for helping me expand my thinking.

Thanks to all the friends who I have gained through fencing, including Parth Deshmukh, Matt Bernstein, Jeffrey Lumish, and Eric Paracka. Their support helped me through the hardest parts of my PhD.

Thanks to Timofey, for his friendship and also for providing me with a different perspective on academia.

Thanks to Sushi Mustwrite, for comforting and enriching my transitioning life at the end of my PhD.

Thanks to Udaya Lakshmi and Shray Bansal, for challenging my mind and helping me cope with the throes of academia.

Thanks to three therapists, who have helped me manage personal growth at different times over the course of my PhD.

Thanks to everyone who helped raise me. I was lucky to have family who gave me love and support throughout childhood and beyond. I was lucky to have friends who made high school not just bearable, but fun and memorable. I was lucky to have friends who pushed me and grew with me throughout undergrad. Grad school was a weird in-between life, but it was easier because it was preceded by a privileged life of friendship, love, and support.

Finally, thanks to my parents, Dan and Margaret Cogswell. More than anyone else, they have provided me with unwavering support throughout my entire life.



They gave me not just freedom, but also support to do what I found interesting,  
even if it included the delayed gratification inherent in a PhD.

# Contents

<b>Acknowledgements</b>	<b>iii</b>
<b>List of Tables</b>	<b>ix</b>
<b>List of Figures</b>	<b>xiv</b>
<b>Summary</b>	<b>xv</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Three Perspectives on Disentanglement and Generalization . . .	4
1.1.1 Disentanglement as Statistical Independence . . . . .	7
1.1.2 Disentanglement as Compositionality . . . . .	9
1.1.3 Disentangling Intention and Language . . . . .	12
1.2 Contributions . . . . .	15
<b>2 DeCov: Decorrelating Hidden Representations</b>	<b>17</b>
2.1 Introduction . . . . .	17
2.2 Approach: DeCov Loss . . . . .	20
2.3 Related Work . . . . .	22
2.4 Experiments . . . . .	25
2.4.1 Dual modality experiments with MNIST: Predicting Side-by-Side Digits . . . . .	25
2.4.2 MNIST Autoencoder . . . . .	27
2.4.3 Image Classification . . . . .	29
2.5 Discussion and Conclusion . . . . .	35

<b>3</b>	<b>Emergence of Compositional Language</b>	
	<b>with Deep Generational Transmission</b>	<b>37</b>
3.1	Introduction . . . . .	37
3.2	Task & Talk: A Testbed for Compositional Language Emer- gence . . . . .	40
3.3	Compositional Language Emergence with Cultural Trans- mission . . . . .	42
3.4	Experimental Setting . . . . .	44
3.5	Results and Analysis . . . . .	46
3.5.1	Impact of Cultural Transmission on Compositional Gen- eralization . . . . .	46
3.5.2	Is Generational Transmission Occurring? . . . . .	48
3.5.3	Visualizing Emergent Languages . . . . .	51
3.6	Related work . . . . .	52
3.7	Conclusion . . . . .	54
<b>4</b>	<b>Dialog Without Dialog: Learning Image Discriminative Di- alog Policies from Single Shot Question Answering Data</b>	<b>56</b>
4.1	Introduction . . . . .	56
4.2	Dialog-based Image Guessing Game . . . . .	59
4.2.1	Game Definition . . . . .	59
4.2.2	Modelling A-bot . . . . .	60
4.2.3	Modelling Q-bot . . . . .	61
4.3	Dialog without Dialog . . . . .	63
4.3.1	Stage 1: Language Pre-training . . . . .	64
4.3.2	Stage 2: Transferring to Dialog . . . . .	65
4.4	Experiments . . . . .	67
4.4.1	Settings . . . . .	67
4.4.2	Metrics . . . . .	68
4.4.3	Results . . . . .	69

4.4.4	Human Studies . . . . .	72
4.4.5	Qualitative Results . . . . .	74
4.4.6	Model Ablations . . . . .	75
4.5	Related Work . . . . .	77
4.6	Conclusion . . . . .	79
<b>5</b>	<b>Conclusion</b>	<b>80</b>
5.1	Implications and Future Work . . . . .	81
5.1.1	Inductive Biases . . . . .	81
5.1.2	Dual Process Theory, Deep Learning, and the Trajectory of AI . . . . .	85
<b>A</b>	<b>Appendix</b>	<b>89</b>
A.1	Appendix for Chapter 2 . . . . .	89
A.1.1	Details of the bias in the MNIST experiment . . . . .	89
A.2	Appendix for Chapter 3 . . . . .	91
A.2.1	Results on Novel Instance Dataset of [Kot+17] . . . . .	91
A.2.2	Replacement Strategies . . . . .	91
A.2.3	Visualization for Language Comparison at Different Training Stages . . . . .	93
A.2.4	Detailed Results . . . . .	94
A.3	Appendix for Chapter 4 . . . . .	99
A.3.1	Architecture Details . . . . .	99
A.3.2	Additional Results . . . . .	102
A.3.3	Mechanical Turk Studies . . . . .	103
	<b>Bibliography</b>	<b>107</b>

# List of Tables

2.1	<b>MNIST side by side results.</b> As expected, biasing right digits at train time so that they are weakly informed by left digits leads to lower performance on an unbiased test set. More importantly, DeCov provides greater improvements over the baselines on the right, confirming that it leads to better features when decorrelation is extremely likely to improve performance. . . . .	26
2.2	CIFAR10 Classification. We can see that DeCov with Dropout leads to the highest test performance and the lowest train-test gap.	29
2.3	CIFAR10 Classification with a bigger version of the base network	30
2.4	CIFAR100 Classification Accuracies . . . . .	31
2.5	ImageNet Classification Accuracies with Network in Network .	34
4.1	Performance of our models and baselines in different experimental settings. From setting A to setting F, agents are tasked with generalizing further from the source data. Our method strikes a balance between guessing game performance and interpretability.	70
4.2	Human evaluation of language quality – question fluency (top) and relevance (bottom). Each row compares a pair of agent-generated questions, asking users which (or possibly neither) is more fluent/relevant. The values report the percentage of times the option represented by that column was chosen. . . . .	72
4.3	Various ablations of our training curriculum. . . . .	76

# List of Figures

1.1	This thesis revolves around three problems: image classification, visual question answering (VQA), and visual dialog. In image classification the goal is simply to pick a descriptive label for an image from a list of choices. The image above might be labeled “fire hydrant” (label not shown). In VQA the goal is to answer a question about an image. Above, one agent asks about fire hydrant color and the VQA agent responds saying the hydrant is red. Finally, in visual dialog agents have a conversation about an image. In the example, the question asking agent follows up the previous question by asking about a related fire hydrant. . . . .	2
1.2	In Section 1.1.1 we consider how to find a function which classifies all of these images correctly. The rightmost images are of different classes, but have similar raw pixel representations, while the leftmost images are both cats, but have less in common. By disentangling representations using statistical independence we want to avoid the tendency of neural nets to overfit to fine-grained patterns like those shared between the cat in Fig. 1.2b and Fig. 1.2c. . . . .	5
1.3	This is a toy example where there are three objects that need to be represented with symbols. Neural networks can do this using one hyperplane per symbol. The right most figure uses one per object, allowing them to be distinguished symbolically. . . . .	9

1.4	Using one hyperplane per example as in the middle figure does not generalize well when a new object like the red square is added to the environment. However, a compositional representation like the one on the right will generalize well by disentangling attributes.	10
1.5	The object reference game of Chapter 3, described in Section 1.1.2.	11
1.6	The image guessing game described in Section 1.1.3. The green bot tries to guess the target image (outlined in red), which only the red bot can see, by asking the red bot questions. These Questions are only answered for the target image, so if they are discriminative enough then they can be used to figure out which image is the target.	14
2.1	Two principal ways to prevent overfitting in deep models are to train with more data (x axis) and to train with Dropout (right plot). As expected, both of these decrease validation error (left axis), but they also happen to decrease hidden activation cross-covariance (right axis). We investigate whether explicitly minimizing cross-covariance can lead to reduced overfitting.	19
2.2	We consider the task of simultaneously predicting two MNIST digits placed side by side. By biasing right digits more than left digits at train time, we create a controlled scenario with the type of problem we expect DeCov to solve.	25
2.3	Weights learned by the first layer of a 2 layer autoencoder are reshaped into images and visualized for a model with no DeCov or Dropout ((Fig. 2.3a)), a model with DeCov ((Fig. 2.3b)), and a model with Dropout ((Fig. 2.3c)).	28

2.4	Cross Entropy and <code>DeCov</code> losses over the course of training AlexNet with 256x256 images. Note that the <code>DeCov</code> val curves are hidden by the train curves. Interestingly, <code>DeCov</code> is reduced even by Dropout, though not nearly as much as when it is explicitly minimized. . . . .	32
2.5	ImageNet classification performance using AlexNet. Plots on the left show training and validation (ILSVRC 2012 validation set) accuracy at different resolutions. Note how all curves have a much lower train-val gap than the (blue) baseline. . . . .	33
3.1	We introduce cultural transmission into language emergence between neural agents. Start with the goal-oriented dialog task at the top of the figure (similar to that of Kottur et al. [Kot+17]). During learning we periodically replace some agents with new ones (gray agents). These new agents do not know any language, but instead of creating one they learn it from older agents. This creates generations of language that become more compositional over time. . . . .	38
3.2	Test set accuracies (with standard deviations) are reported against our new harder dataset using models similar to those in [Kot+17]. Our variations on cultural transmission (darker blue bars) outperform the baselines without cultural transmission. . . . .	46
3.3	Do bots in a population learn similar languages? On the y-axis (eq. (3.2)) lower values indicate more similar language. Bots from our method speak similar languages, but independently evolved agents do not. Thus our implicit procedure induces cultural transmission. . . . .	50



3.4	All conversations between Q-bot 4 and Q-bot 3 for the ( <code>shape</code> , <code>color</code> ) task. These bots were trained in the <i>Multi Agent Oldest</i> setting. The figure shows A-bot's utterances for each object and whether or not Q-bot guessed the object correctly, as described in Section 3.5.3. This language is compositional because each token refers to a color or shape. . . . .	51
4.1	(Top - 2 pools) We train our questioner to ask questions that can discriminate between pairs of images by mimicing questions from the VQAv2 dataset. (Bottom - 1 pool) Our proposed model generalizes to new settings in a way that humans can understand without additional language supervision ( <i>i.e.</i> , without dialog). . .	57
4.2	A single round of our Q-bot which decomposes into the modules described in Section 4.2.3. This factorization allows us to fine-tune just the intention of the model for task performance, limiting language drift. . . . .	61
4.3	Qualitative comparison of dialogs generated by our model with those generated by Non-Var Cont and Stage 1 baselines. Top / middle /bottom rows are image pool from COCO / AWA / CUB images respectively. Our model pretrained on VQA (COCO image) generates more interpretable questions for the DwD task which is semantic meaning and generalize well to out-of-domain images. . . . .	74
5.1	As described in Fig. 1.4, Fig. 5.1a represents objects in a one hot fashion and Fig. 5.1b represents objects as compositions of attributes. In both figures the training data (the triangles and the blue square) can be perfectly distinguished from one another, but they do not generalize equally well to the test data (red square). An inductive bias like the population dynamics of Chapter 3 is needed to pick representations like the one in Fig. 5.1b. . . . .	82

A.1	Test set accuracies (with standard deviations) are reported by training and evaluating the same models as in our main results (figure 2 main paper) against the dataset from [Kot+17]. These results do not perform cross-validation, following [Kot+17]. They only vary across 4 different random seeds. See Section A.2.1.	92
A.2	Each sub-figure summarizes an A-bot’s language, as described in section 5.3 of the main paper. By comparing the baseline of (Fig. A.2a) to a similar pair of bots from our approach (Fig. A.2b) we can see that our approach encourages compositional language to emerge. Furthermore, the similarity between (Fig. A.2b) and (Fig. A.2c) suggests language is indeed transmitted in our approach.	95
A.3	Replacement strategy comparison p-values.	96
A.4	Single Agent model comparison p-values.	97
A.5	Multi Agent model comparison p-values.	98
A.6	Task performance (guessing game accuracy) over rounds of dialog. Performance increases over rounds for all models except the Stage 1 models.	103
A.7	An example of the interface used to ask humans to evaluate question relevance.	104
A.8	An example of the interface used to ask humans to evaluate question fluency.	105
A.9	An example of the interface subjects used to interact with our Q-bot models.	106

# Summary

Despite the increasingly broad perceptual capabilities of neural networks, applying them to new tasks requires significant engineering effort in data collection and model design. Generally, inductive biases can make this process easier by leveraging knowledge about the world to guide neural network design. One such inductive bias is disentanglement, which can help prevent neural networks from learning representations that capture spurious patterns that do not generalize past the training data, and instead encourage them to capture factors of variation that explain the data generally.

In this thesis we identify three kinds of disentanglement, implement a strategy for enforcing disentanglement in each case, and show that more general representations result. These perspectives treat disentanglement as statistical independence of features in image classification, language compositionality in goal driven dialog, and latent intention priors in visual dialog. By increasing the generality of neural networks through disentanglement we hope to reduce the effort required to apply neural networks to new tasks and highlight the role of inductive biases like disentanglement in neural network design.

# Introduction

Deep Learning (DL) has been an exciting new development for Artificial Intelligence (AI). It has led to great progress on problems that are fundamentally perceptual and thus traditionally hard. Now the field can classify images [KSH12; He+16; SZ14; Xie+16], meaningfully relate text to images [Ant+15; Jia+18; And+18; Lu+19; Xu+15], and learn to play difficult games [Sil+16; Mni+15]. These models even outperform humans sometimes [DK17; Sil+17]. This is enabled by Neural Networks (NNs) that shortcut difficult or ill posed logical problem by instead recognizing patterns. However, as we approach increasingly complex problems we are still challenged to scale AI with that complexity.

To see this challenge compare image classification, visual question answering, and visual dialog. We know how to train an image classifier to accurately label images with one of 1000 classes [Rus+14; He+16]. First collect lots of labeled examples (*e.g.*, 10,000 per class), then train a convolutional neural net to predict the correct labels. The neural net will learn features that represent image semantics that are more directly useful for classification than pixel intensities. Large scale labeled image data is expensive to collect and neural nets can be difficult to design, but the approach achieves accuracy on par with humans.<sup>1</sup>

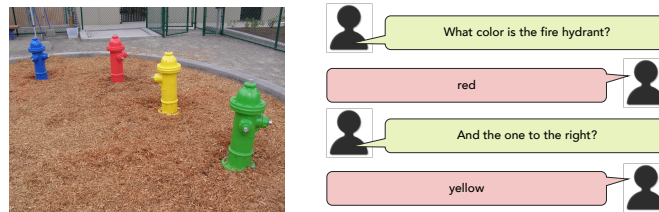
In visual question answering (VQA) the goal is to answer a question (*e.g.*, “What color is the animal?” in Fig. 1.1) about an image instead of simply labeling that image. This scales complexity by requiring VQA models to understand

---

<sup>1</sup>Note the task is constrained to the distribution of images and classes found in ImageNet [Rus+14].

natural language in addition to the visual world. As before, the approach is to collect a large dataset of examples [Ant+15] – (image, question, answer) tuples in this case – and train a neural net to predict the answer given the image and question [Jia+18]. The VQA model needs to be able to understand much of the same visual content as the image classifier, so instead of learning a good image representation from scratch it directly uses the representation learned by the image classifier. Nonetheless, it still needs to learn to represent language and to ground that language in the classifier’s visual representation.

But this time the approach is less effective, achieving quantifiably and qualitatively worse performance than humans [Ant+15]. Even the best VQA models [Jia+18] that take this primarily data-driven approach fail at simple tasks like counting objects in an image. Moreover, data is more expensive to collect because it requires generating not just answers to questions, but also the questions themselves. At the same time the space of possible examples (image, question, answer tuples) is larger since there are more questions one might pose in natural language than the 1000 labels from ImageNet [Rus+14].



**Fig. 1.1.:** This thesis revolves around three problems: image classification, visual question answering (VQA), and visual dialog. In image classification the goal is simply to pick a descriptive label for an image from a list of choices. The image above might be labeled “fire hydrant” (label not shown). In VQA the goal is to answer a question about an image. Above, one agent asks about fire hydrant color and the VQA agent responds saying the hydrant is red. Finally, in visual dialog agents have a conversation about an image. In the example, the question asking agent follows up the previous question by asking about a related fire hydrant.

A similar story holds for visual dialog, which allows a sequence of questions like the two in Fig. 1.1. The second question in Fig. 1.1 (about the yellow

hydrant) depends on the previous questions and answers, making the task more complex and requiring a representation of dialog history to be learned. Humans answer questions much better than neural nets trained with lots of examples [Das+17b] and data collection is more expensive than previous tasks, mirroring the challenges of VQA.

Representation learning is key in all these cases, but it also becomes less effective as tasks get more complex. To address these issues we will continue to focus on representation learning, but look for ways to improve it in addition to just collecting more data. This will be done by using our prior knowledge about the tasks and the world to bias the representations these neural nets learn.

In particular we focus on *disentangled representations*, and show that by encouraging neural net representations to be disentangled they perform better in terms of generalization to new examples and new tasks. Though neural nets learn useful representations, learning is terribly underconstrained, leading to concepts that are not very general [Zha+16] and shifting the attention of designers to the inductive biases we bake into these models. Disentanglement is one such inductive bias. In its most general – hence not very useful – sense, disentanglement is the idea that representation learning can be constrained by forcing different factors of variation to be represented separately [BCV13]. A representation thus constrained should generalize better.

This thesis continues by describing three different perspectives on disentanglement and relating them to existing work. It then devotes three chapters – Chapter 2, Chapter 3, and Chapter 4 – to detailed explanations of how these perspectives on disentanglement can be implemented and how doing so increases the generalizability of the corresponding representations. The final chapter concludes by summarizing our work and reflecting on the role of disentanglement and inductive biases more generally in neural net design.

## 1.1 Three Perspectives on Disentanglement and Generalization

In this work *disentanglement* has three somewhat different definitions, each paired with a slightly different notion of generalization. For each case we provide a concrete definition of what disentanglement is and in the body of the thesis we verify that by encouraging disentanglement we can improve generalization. This section first offers a common framework for thinking about these notions of disentanglement, then details each of the three perspectives in individual sub-sections.

Disentanglement is only useful because of the structure we observe in the world, so we have to start by understanding that structure. Only certain scenes can be, or are likely to be, physically realized. Because of this, we can often summarize our observations using a small amount of information communicated as a few salient factors of variation. For example, “a black cat on a white background” is a small amount of information summarizing Fig. 1.2b. It only applies to very few of all the possible images of cats, much less all possible images. With more information about what task this representation is supposed to help perform, *e.g.* object classification, we can further reduce the summary to simply “a black cat.”

In general, the world contains direct observations formalized as input vectors  $\mathbf{x}$  (usually a vector of real numbers). They are typically represented by some neural network  $f$  as vectors  $\mathbf{h} = f(\mathbf{x})$ . There are also true but unknown “generative factors”, dimensions of some vector  $\mathbf{z}$ , such that all the variations of  $\mathbf{z}$  correspond to some  $\mathbf{x}$  that occurs in the real world, though it might be that



**Fig. 1.2.:** In Section 1.1.1 we consider how to find a function which classifies all of these images correctly. The rightmost images are of different classes, but have similar raw pixel representations, while the leftmost images are both cats, but have less in common. By disentangling representations using statistical independence we want to avoid the tendency of neural nets to overfit to fine-grained patterns like those shared between the cat in Fig. 1.2b and Fig. 1.2c.

some variations in  $\mathbf{x}$  are not likely to be generated by any  $\mathbf{z}$ . If  $\mathbf{h}$  is a disentangled representation then its dimensions will correspond to the dimensions of  $\mathbf{z}$ .

To take some intuitive examples, the latent  $\mathbf{z}$  might correspond to some combination of things like the rotation of digits / faces, the classes objects fall into, or attributes like color and shape. If we can get neural nets to discover generalizable representations like these, then those representations will support better performance on the tasks they are used for.

This setup is fairly vague. What does it mean for the dimensions of  $\mathbf{h}$  to “correspond” to the dimensions of  $\mathbf{z}$ , and more importantly what are these factors of variation  $\mathbf{z}$  exactly? There is not a good general answer because  $\mathbf{z}$  depends strongly on domain specific knowledge, and the choice of factors might not even be unique. Each choice of  $\mathbf{z}$  effectively defines an implicit task which different disentanglement priors may be more or less well adapted to. This ambiguity makes it useful to define different notions of disentanglement, based on different domain specific intuitions about the latent factors. The subsequent



sub-sections do exactly this, but before proceeding we will briefly summarize the place of disentanglement in the deep learning literature.

The idea that there is low-dimensional structure to the world, a manifold, and that neural networks should take advantage of it is fundamental to representation learning [LBH15; BGV13; GBC16]. As a result, it has been explored extensively in literature on unsupervised learning of disentangled representations [Hig+17; Kul+15; DB17]. If neural nets start with disentangled representations before being trained to accomplish goals then they may be able to accomplish those goals more efficiently or more accurately.

A number of works have taken up this idea under the label of *unsupervised* pre-training and observed increased interpretability and better generalization (mainly using synthetic data) of the resulting representations [Hig+17; Che+16; Kul+15]. Some work has even applied these methods to additional tasks and found improved generalization Steenkiste et al. [Ste+19] and Esmacili et al. [Esm+18]. What makes this possible in an unsupervised is that the inductive biases of disentanglement are compatible with the inductive biases found in the data [Loc+19]. On the other hand, supervision allows the inductive bias of a particular task to strongly inform the model and which factors are useful to disentangle for the task at hand.

This brings us to the proposed notions of disentanglement. In each of the three sections below we describe a problem and how a more specific notion of disentanglement can be used to solve it.

### 1.1.1 Disentanglement as Statistical Independence

In Chapter 2 a disentangled representation  $\mathbf{h}$  is one where changes in one (or a few) dimensions of  $\mathbf{z}$  correspond to changes in one (or a few) dimensions of  $\mathbf{h}$ . Intuitively, this sounds like statistical independence, so in this perspective the more statistically independent the dimensions of  $\mathbf{h}$ , the more disentangled the representation. Redundant representations might capture spurious patterns and overfit to fine-grained differences, like those between Fig. 1.2b and Fig. 1.2c, as opposed to the more general patterns shared by the cats but not the dog in Fig. 1.2.

We penalize redundant representations when training image classifiers. Instead of optimizing just the cross-entropy loss  $\mathcal{L}_{CE}$  commonly used for training neural network classifiers we penalize redundancy with an additional term  $\mathcal{L}_{DeCov}$  weighted by hyperparameter  $\lambda$ :

$$\mathcal{L} = \mathcal{L}_{CE} + \lambda \mathcal{L}_{DeCov} \quad (1.1)$$

Given activation values  $h_i^n$  and  $h_j^n$  of neurons  $i$  and  $j$  for example  $n$ , we measure the covariance between  $i$  and  $j$  using the current (size  $N$ ) batch of examples

$$C_{i,j} = \frac{1}{N} \sum_{n=1}^N (h_i^n - \mu_i)(h_j^n - \mu_j) \quad (1.2)$$

where  $\mu_i$  is the empirical mean of activation  $i$  over the batch. This allows us to use the Frobenius norm  $\|\cdot\|_F$  to aggregate all covariances between different neurons into  $\mathcal{L}_{\text{DeCov}}$

$$\mathcal{L}_{\text{DeCov}} = \frac{1}{2} \left( \|C\|_F^2 - \|\text{diag}(C)\|_2^2 \right) \quad (1.3)$$

This loss, described in detail in Chapter 2, is larger for redundant representations, so penalizing it minimizes redundancy.

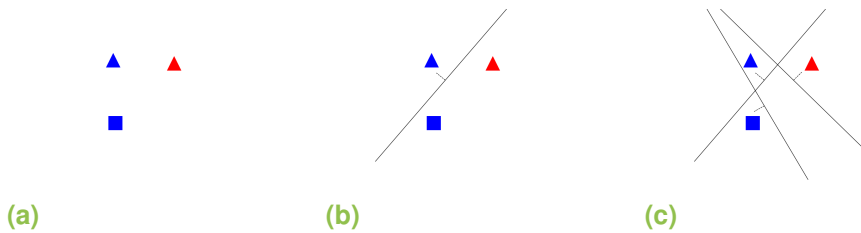
Training with this loss reduces overfitting in large CNN image classifiers. To measure this we use test accuracy and the gap between train and test accuracy, computing both metrics for a number of image classification CNNs and datasets. Generally, test accuracy increases and the gap between train and test accuracy decreases when we use the DeCov loss. By disentangling representations with the DeCov loss we increase the generalization capabilities of CNN image classifiers.

This view of disentanglement as statistical independence is also popular in the unsupervised disentanglement literature. There statistically independent representations form the basis of some of the most important work like  $\beta$ -VAE [Hig+17], where independence results from a variational prior. Further related work directly penalizes the total covariance of  $\mathbf{h}$  [KSB17] in a fashion similar to that described in Chapter 2. Both of these works succeeded the work of Chapter 2.

### 1.1.2 Disentanglement as Compositionality

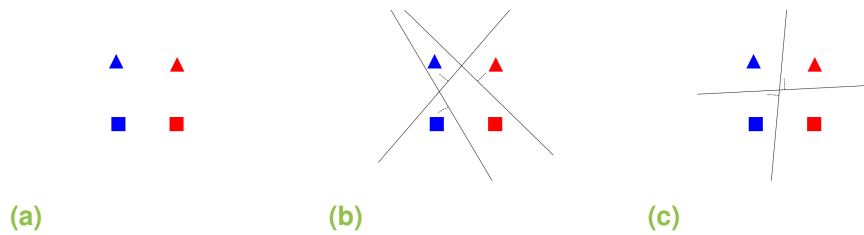
In Chapter 3 we treat disentanglement as compositionality in language. Eventually we want systems that use neural networks to interact with humans using language, so they need to understand symbolic representations like words. A key feature of these types of representation is compositionality, when words that themselves have meaning can be combined with other words to form new meanings.

To see why this is important and how it is an example of disentanglement, consider how a neural net might represent words for each of the objects in Fig. 1.3a. These objects are embedded into a continuous vector space (here the 2d plane) and we want to do a simple classification task, assigning a word to each object. Here we take a geometric perspective on how neural nets do this, by using one hyperplane per word [Mon+14]. For example, the blue triangle might be represented by the black hyperplane in Fig. 1.3b (the dashed line is the normal vector) and then the other two objects by the additional hyperplanes in Fig. 1.3c. The hyperplane an object is farthest in front of corresponds to the word used by the neural net to represent that object. This toy example makes it fairly easy to place one hyperplane per object and thereby represent each of the objects from the available set.



**Fig. 1.3.:** This is a toy example where there are three objects that need to be represented with symbols. Neural networks can do this using one hyperplane per symbol. The right most figure uses one per object, allowing them to be distinguished symbolically.

But what happens when we get a new object we have not seen before? Because the world often has compositional structure, this new object is likely to be similar to past observations but with a novel composition of attributes, like the red square from Fig. 1.4a. Using the previous hyperplanes the red square is going to be represented with the same word as the red triangle since it is closest to being in front of that hyperplane (it has the highest activation when projected on to that normal vector). This prevents the model from distinguishing between red shapes (Fig. 1.4b).

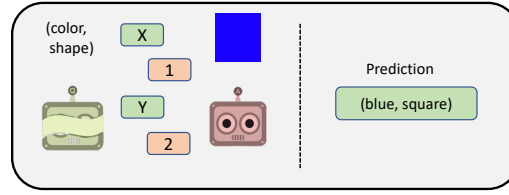


**Fig. 1.4.:** Using one hyperplane per example as in the middle figure does not generalize well when a new object like the red square is added to the environment. However, a compositional representation like the one on the right will generalize well by disentangling attributes.

However, an alternate representation could use compositional features like the shape and color attributes and it would be able to distinguish between red shapes. Such a representation disentangles the two attributes and thus generalizes better. One way for a neural net to do this would be to use two words corresponding to the two hyperplanes from Fig. 1.4c – one for color and one for shape.

In Chapter 3 we study compositionality using the simple object reference game described in Fig. 1.5. There are two agents, Q-bot (in green) and A-bot (in red). A-bot gets to see an object (in this case, a blue square) that Q-bot does not know about. Q-bot is told to predict two attributes of the object (here, color and shape), and the two bots communicate with each other to pass on information about the object. Ideally, the dialog looks something like the one in Fig. 1.5. Q-bot asks "X" (what color?), A-bot responds "1" (blue), Q-bot asks "Y" (what shape?),

and A-bot responds "2" (square). After the dialog Q-bot makes a prediction about the color and shape of the object (blue, square).



**Fig. 1.5.:** The object reference game of Chapter 3, described in Section 1.1.2.

In this toy setting it is trivial to supervise Q-bot and A-bot with a compositional language, designed by humans, that disentangles the various attributes of interest, but we are interested in studying how a compositional language might emerge without this supervision. We are interested in what it takes for these agents to discover a compositional language from feedback about task performance. Thus we reward both agents when Q-bot guesses both attributes correctly, and we give no reward otherwise. A language that can communicate object attributes emerges as a result of this feedback, but it is not necessarily compositional. When a new object is shown to A-bot, it won't necessarily be able to communicate both attributes effectively because it may have learned a language more like the one in Fig. 1.4b than the one from Fig. 1.4c.

We improve this ability to learn compositional language by adding a cultural transmission mechanism to train our bots with. In evolutionary linguistics cultural transmission has been shown to increase the compositionality of language [Kir01; KCS08; KGS14]. These studies simulate the transmission of language between pairs of agents in a sequence. Agent A is given a random language and teaches part of it to agent B. This gives agent B a new language, which it then teaches to agent C. As agents continue to learn in this fashion the language itself changes to become more compositional than it was at first.

We transfer this result to our object reference game with neural agents using a more indirect approach which implicitly encourages transmission of language between agents and still increases language compositionality. To achieve implicit language transmission we 1) create a population of many Q-bots and many A-bots, and 2) create dynamics in the population by replacing some agents periodically. Each new agent has an ineffective (randomly generated) language. This creates a knowledge gap that encourages these new agents to learn from the old agents that already know an effective language, implicitly mimicing the transmission protocol from before. In Chapter 3 we show that this encourages the resulting languages to generalize to unseen objects like the red square from Fig. 1.4a.

### 1.1.3 Disentangling Intention and Language

In Chapter 4 our final perspective focuses on scaling question representations to goal driven visual dialog by transferring language information from VQA. Our goal driven dialog setting requires agents to ask questions about images, like in VQA, but for a different reason. Thus by learning to generate the words in a question (language) and then learning what that question should be about (intention) we can efficiently learn a representation that transfers the needed information while being flexible enough to adapt to a new task.

We study this problem using an image guessing game similar to the object reference game from Chapter 3, but with a vision component (natural images) and natural language (English). In this goal driven visual dialog setting we provide the agents with an image guessing game to solve. Two agents are presented with the pool of images shown on the left of Fig. 1.6. One is identified as the target image (number 3 in Fig. 1.6), and this information is only revealed to the question answering agent A-bot. Next the bots engage in a couple rounds

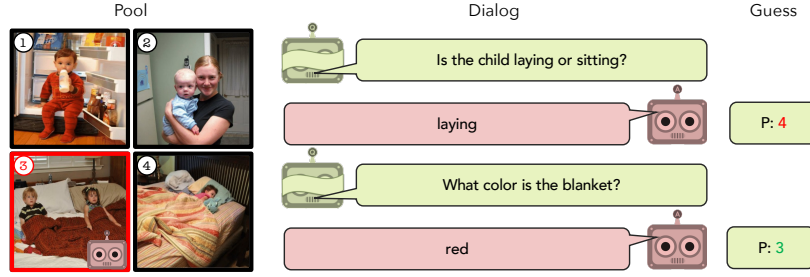
of dialog. In each round Q-bot asks a question about the pool of images and A-bot answers it with respect to only the target image. Then Q-bot makes a guess about which image is the target given only the dialog so far and the image pool. Thus Q-bot asks questions to help it find the target image. In this context, a good question allows A-bot’s answer to discriminate between images in the pool instead of trying to “stump” A-bot, as in the original question motivation for VQA. The questions in VQA are similar to those in visual dialog, but their intent is different because the two tasks have different goals.

We could approach this problem by trying to collect complete dialogs where participants play the image guessing game, then train Q-bot to generate those questions. This would align the intent and language of the questions, but it would be expensive and it wouldn’t scale to new tasks; every new task would require a new dataset.

An alternate approach, which we take in Chapter 4, is to transfer language to the image guessing game from an existing dataset (VQA) and then learn to solve the new task by rewarding Q-bot when it guesses the target correctly. This task level feedback is easy to compute because we know the target image. To generalize from VQA to goal driven dialog we design a model that first determines question intention and realizes that intention in a sequence of words.

Agents’ intentions depend on their goals, so we must discuss what agents are trying to accomplish when they ask questions in VQA and visual dialog. In the VQA dataset the motivation is to develop a question answering agent and not a question asking agent, so questions are asked by humans. Humans were told to ask questions that would stump a smart robot [Ant+15]. This task is different from the one we ask our visual dialog agents to perform (to guess the target image).





**Fig. 1.6.:** The image guessing game described in Section 1.1.3. The green bot tries to guess the target image (outlined in red), which only the red bot can see, by asking the red bot questions. These Questions are only answered for the target image, so if they are discriminative enough then they can be used to figure out which image is the target.

Though the intention is different in the two tasks, questions need be expressed in natural language in both cases, so both questioner agents need an understanding of how to generate valid questions. Normally this can be accomplished by training Q-bot to mimic human generated questions *for the relevant task*, like those generated for VQA [Mos+16]. But for more complex tasks like our goal driven visual dialog task this approach becomes less appealing. The increased complexity of visual dialog makes it more data hungry and at the same time more expensive to collect data for.

We approach this with transfer learning, by leveraging data already collected for VQA to solve the more complex visual dialog task. Questions generated for VQA and visual dialog are similar in that they both must be expressed in valid language (*i.e.*, English in this case), but different in their intention (they have different goals). This suggests we should transfer knowledge about language from VQA to visual dialog, but ignore the intention of the VQA questions.

In particular, Q-bot first generates a latent variable constrained to represent intention and then generates language conditioned on this latent variable. The language generation part of Q-bot is trained to mimic human language from the VQA dataset but is not trained to solve the image guessing game. As Q-bot

is trained for the image guessing game discussed above it can only adapt its intention representation.

We show that without these steps to promote disentanglement Q-bot is indeed able to guess the target image correctly. However, even though it was initialized with some knowledge of what English questions look like, we show that its language drifts and no longer looks like valid English when disentanglement is not used. With disentanglement we show that the language looks a lot more like valid English to humans who judge it. Moreover, it acts like valid English, because humans can understand it well enough for Q-bot to guess the right image.

## 1.2 Contributions

- In chapter Chapter 2 we show how disentangling latent variables using statistical independence at a low level makes NNs generalize better to new examples from the same domain. Disentanglement is realized via an additional regularizing loss which penalizes redundancy or statistical dependence.
- In chapter Chapter 3 we cast language in goal oriented dialog as a latent variable and show how to make it compositional without direct supervision. In particular, adding multiple agents with cultural dynamics encourages compositionality. As we point out, this is a commonly known effect in Evolutionary Linguistics, but we are the first to apply the idea to neural networks. We show that the languages, which *emerge* as a result of goal driven behavior, generalize compositionally.

- In Chapter 4 we focus on a more realistic visual dialog setting where the language is English and the task involves natural images. We discuss how to build a question asking agent that disentangles language from intention. Our agent not only solve new tasks effectively, but it does so without forgetting what valid English looks like. This allows our agent to interact with humans to solve tasks it did not receive language supervision for.

In summary, this thesis identifies the importance of the disentanglement bias to generalization in representation learning by describing three notions of disentanglement and showing how disentanglement promotes generalization in each case.

# DeCov: Decorrelating Hidden Representations

## 2.1 Introduction

Deep Neural Networks (DNNs) have recently achieved remarkable success on a wide range of tasks – *e.g.*, image classification on ImageNet [KSH12], scene recognition on MIT Places [Zho+14], image captioning with MS COCO [Lin+14a; Vin+14; CZ15], and visual question answering [Ant+15]. One significant reason for improvement of these methods over their predecessors has to do with scale. Faster computers coupled with optimization improvements such Batch Normalization, Adaptive SGD, and ReLus let us quickly train wider and deep networks. Access to large annotated datasets and regularizers such as Dropout has provided significant reduction in the amount of overfitting in these large networks, thus enabling the performance we see today.

In this paper, we focus on the problem of overfitting, which is observed when a high capacity model (such as a DNN) performs very well on training data but poorly on held out data. Even when trained on large annotated datasets (such as ImageNet [Rus+14] or Places [Zho+14], containing millions of labelled images), deep networks are susceptible to overfitting. This problem is further exacerbated when moving to new domains and tasks – since DNNs tend not to generalize with a few examples, each new task tends to require curating and

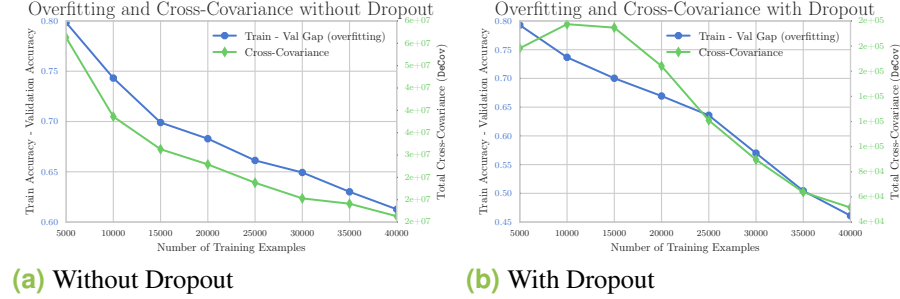
annotating a new large dataset. While there has been some success with transfer learning [Gir+14; Don+14; Yos+14], networks still overfit.

A promising alternative to creating even larger datasets is to apply different forms of regularization to the network while training to avoid overfitting. These methods include regularizing the norm of the weights [Tik43], Lasso [Tib96], Dropout [Sri+14], DropConnect [Wan+13], Maxout [Goo+13], etc.

One particular regularizer of interest to DNNs is Dropout [Sri+14], which attempts to prevent co-adaptation of neuron activations. Co-adaptation occurs when two or more hidden units rely on one another to perform some function which helps fit training data, thus becoming highly correlated. Co-adaptation is reduced by Dropout using an approximate model averaging technique that sets a randomly selected set of activations to zero at training time. [Sri+14] show that this has a regularizing effect, leading to increased generalization and sparser, less correlated features. Notice that this is without *explicitly* encouraging decorrelation in hidden activations.

To further investigate the relationship between hidden activation correlations and overfitting, we show in (Fig. 2.1) two quantities from a CNN trained for image classification on CIFAR100 [KH09] – (1) the amount of overfitting in the model (as measured by the gap between train and val accuracy), and (2) the amount of correlation in hidden activations (as measured by the Frobenius norm of the sample cross-covariance matrix computed from vectors of hidden activations; details in Section 2.2). Both these quantities of interest are reported as a function of amount of training data (x-axis) and with/without Dropout (left/right subplot). As expected, both increased training data and Dropout have a regularizing effect and lead to reduced overfitting.

The figure also shows an interesting novel trend – as the amount of overfitting reduces, so does the degree of correlation in hidden activations. In essence, overfitting and co-adaptation seem to be correlated. The open question of course is – is the relationship causal?



**Fig. 2.1.:** Two principal ways to prevent overfitting in deep models are to train with more data (x axis) and to train with Dropout (right plot). As expected, both of these decrease validation error (left axis), but they also happen to decrease hidden activation cross-covariance (right axis). We investigate whether explicitly minimizing cross-covariance can lead to reduced overfitting.

This leads to the principal questions of this paper – Is it possible to bias networks towards decorrelated representations by directly reducing correlation between hidden units? And do such decorrelated representations generalize better?

**Overview and Contributions.** The goal of this paper is to learn DNNs with decorrelated activations and study the effect of this decorrelation on their generalization performance. Towards this end, we propose a fairly natural loss called DeCov, which explicitly encourages decorrelation between the activations in a deep neural network. This loss requires no additional supervision, so it can be added to any existing network.

In addition to the link discussed above, our motivation also comes from the classical literature on bagging and ensemble averaging [HS90; PC93; Bre96], which suggests that decorrelated ensembles perform better than correlated ones.

Our experiments encompass a range of datasets (MNIST [LeC+95], CIFAR10/100 [KH09], ImageNet [Rus+14]), and different kinds of network architectures (Caffe implementations of LeNet [LeC+95], AlexNet [KSH12], and Network in Network [LCY13]). All cases suggest that DeCov acts as a novel and useful regularizer.

## 2.2 Approach: DeCov Loss

To express our notion of redundant or co-adapted features, we impose a loss on the activations of a chosen hidden layer. In a manner similar to Dropout, our proposed Decov loss may be applied to a single layer or multiple layers in a network. For simplicity, let us focus on a single layer. Let  $\mathbf{h}^n \in \mathbb{R}^d$  denote the activations at the chosen hidden layer, where  $n \in \{1, \dots, N\}$  indexes one example from a batch of size  $N$ . The covariances between all pairs of activations  $i$  and  $j$  form a matrix  $C$ :

$$C_{i,j} = \frac{1}{N} \sum_n (h_i^n - \mu_i)(h_j^n - \mu_j) \quad (2.1)$$

where  $\mu_i = \frac{1}{N} \sum_n h_i^n$  is the sample mean of activation  $i$  over the batch.

We want to minimize covariance between different features, which corresponds to penalizing the norm of  $C$ . However, the diagonal of  $C$  contains the variance of each hidden activation and we have no reason to require the dynamic range of activations to be small, so we subtract this term from the matrix norm to get our final DeCov loss

$$\mathcal{L}_{\text{DeCov}} = \frac{1}{2} \left( \|C\|_F^2 - \|\text{diag}(C)\|_2^2 \right) \quad (2.2)$$

where  $\|\cdot\|_F$  is the frobenius norm, and the  $diag(\cdot)$  operator extracts the main diagonal of a matrix into a vector. In our experiments, subtracting the diagonal made little difference for small networks, but led to increased stability for larger networks.

Perhaps the best quality of this loss is that it requires no supervision, so it can be added to any set of activations. In a manner similar to Dropout, our experiments typically apply Decov loss to fully connected layers towards the deep end of a network (*e.g.*, fc6 and fc7 for AlexNet). However, note that Decov affects *all parameters* up to the layer where it is applied (and not just the parameters in the specific layer).

At first glance, one seeming peculiarity about this loss is that its global minimum can be found by setting all weights for  $\mathbf{h}$  to 0. This is similar to an  $L_2$  regularizer in that both encourage weights to tend toward 0, but one important difference between these two regularizers is that  $\mathcal{L}_{\text{DeCov}}$  depends on input data and is not a function purely of a weight vector like one might find in a classical regularizer such as  $L_2$  or  $L_1$ .

To understand this further, consider the gradient of the loss with respect to a particular activation  $a$  for a particular example  $m$

$$\frac{\partial \mathcal{L}_{\text{DeCov}}}{\partial h_a^m} = \frac{1}{N} \sum_{j \neq a} \left[ \frac{1}{N} \sum_n (h_a^n - \mu_a)(h_j^n - \mu_j) \right] (h_j^m - \mu_j). \quad (2.3)$$

Let us denote the rightmost term in this expression by  $I(j, m) = (h_j^m - \mu_j)$ .

This term is large (in absolute value) when feature  $j$  is discriminative for example  $m$  w.r.t. the mean of the batch. If  $j$  were not discriminative for  $m$  then  $h_j^m$  would be close to  $\mu_j$ . Hence, we can consider  $I$  as an “importance” term, corresponding to a notion of how significant feature  $j$  is for example  $m$ .



Also notice that the term on the left in the gradient expression is simply the covariance between feature  $a$  and feature  $j$ . Thus, the gradient can be re-written as

$$\frac{\partial \mathcal{L}_{\text{DeCov}}}{\partial h_a^m} = \frac{1}{N} \sum_{j \neq a} C_{a,j} \cdot I(j, m). \quad (2.4)$$

**Interpretation.** Intuitively, the covariance term can be thought of as measuring (linear) redundancy: features  $a$  and  $j$  are redundant if they vary together. Thus, the `DeCov` loss tries to prevent features from being redundant, but redundancy is weighted by importance ( $I$ ). Specifically, a feature  $j$  contributes towards a large gradient of feature  $a$  on example  $m$  if  $j$  is important for  $m$  and correlated with  $a$ . This means important features correlated with  $a$  (e.g.,  $j$ ) contribute to a large gradient of  $a$ , suppressing the activation  $h_a^m$ . A feature which fires only in specialized situations (e.g., a cat’s ear) will likely be nearly identical or noisy for most other examples (e.g., non-cats) and will not contribute towards gradients of other specialized features.

## 2.3 Related Work

**Redundancy Based Representations.** The idea of using low redundancy to learn representations has been around for decades. In an early attempt to model human perception, [Bar61] lists 3 possible learning principles, the 3rd being the notion that representations should not be redundant.

Later work continued to investigate this intuition in the context of unsupervised feature learning. Three objectives emerged, each of which formalize the notion differently. (1) An information theoretic view is expressed by [Lin88]. The main idea is to maximize information gained by predicting the next rep-

resentation/layer between input and output. (2) The closest objective to ours is cross-correlation (not cross-covariance), which appears in [BB09] and complements a temporal coherence objective. It also appears in [PH86] where it complements an objective which encourages units to capture higher order input statistics. (3) Finally, redundancy minimization is realized through predictability minimization in [Sch92] for the purpose of learning factorial codes (representations whose units are independent). This objective says that one unit should not be predictable given *all* of the others in its layer as input.

All of these works focus on unsupervised feature learning and do not experiment with supervised models. Furthermore, these early pioneering works were limited by data and evaluated small networks without many of the modern design choices and features (e.g. ReLus, Dropout, SGD instead of Hebb’s update rule, batch-normalization, *etc.*). We propose redundancy minimization for a new purpose (regularization), evaluate it using modern techniques such as end-to-end learning using SGD with respect to a supervised objective, and do this in the context of harder challenges presented by modern datasets. To the best of our knowledge, such a setting has not been considered before.

**Correlation/Covariance Losses in Other Settings.** Other works have used similar penalties, but in different settings and to different effects. Deep Canonical Correlation Analysis (Deep CCA) [And+13] and Correlational Neural Networks (CorrNets) [Cha+15] apply a similar loss which *maximizes* correlation, unlike our *minimization* of cross-covariance. Both methods are used to learn better features in the presence of multiple views or modalities. They embed inputs to a common space and maximize correlation between aligned pairs.

Another idea similar to ours is that of [Che+14], which aims to discover and disentangle hidden factors. The goal is to separate supervised factors of variation (*e.g.*, class of MNIST digits) from unsupervised factors of variation (*e.g.*, handwriting style). In order to achieve this goal, they impose a covariance (not correlation) loss between (1) the softmax outputs of a neural network trained to recognize digits and (2) a hidden representation which is used in conjunction with (1) to reconstruct the input (via an auto-encoder).

These two works suggest that correlation losses significantly impact learned representations in the context of modern networks. One key difference between these two approaches and ours is that while their formulations decorrelate [Che+14] and disregard [And+13; Cha+15] parts of *different* representations, our approach tries to decorrelate parts of the *same* representation. Moreover, the ultimate goals are different. Unlike these approaches, our goal is simply to improve supervised classification performance by reducing overfitting, and not to reconstruct the original data.

**Dropout and Batch Normalization.** Two recent approaches to regularization in deep neural networks are Dropout [Sri+14] and to some extent Batch Normalization [IS15]. Dropout aligns with our intuition and goals more closely as it aims to improve classification performance by reducing co-adaptation of activations. On the other hand, Batch Normalization focuses on faster optimization by reducing *internal co-variate shift*, which is the constant variation of a layer’s input as it learns. Some Batch Normalization results indicate it could act as a regularizer, but this has not been exhaustively verified yet. Our approach is similar to Batch Normalization due to its use of mini-batch statistics.

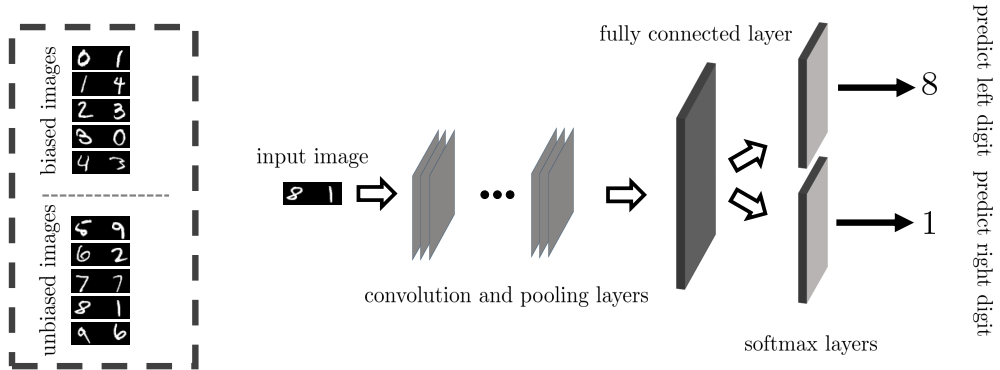
## 2.4 Experiments

We begin with a synthetic dual “modality” experiment, which serves as a testbed for measuring improvement due to decorrelation. Next, we use an autoencoder (as in [Sri+14]) to contrast DeCov and Dropout. Finally, we use a variety of experiments to report Image Classification performance on CIFAR10/100 and ImageNet, noticing significant improvement in *all cases*. Note that we set the Dropout rate to 0.5 as suggested by [Sri+14].

### 2.4.1 Dual modality experiments with MNIST:

#### Predicting Side-by-Side Digits

We propose a synthetic dual “modality” task on MNIST – simultaneously predict the class labels for two digits placed adjacent in an image. We created a dataset where each example consists of two MNIST digit images horizontally concatenated and separated by 16 black pixels (to prevent interference between feature maps in the first layers). (Fig. 2.2) shows a few examples.



**Fig. 2.2.:** We consider the task of simultaneously predicting two MNIST digits placed side by side. By biasing right digits more than left digits at train time, we create a controlled scenario with the type of problem we expect DeCov to solve.

The important detail of this experiment is the particular bias we inject into the distribution of left and right digits. Let

$$P(l) = 0.1 \text{ and } P(r|l) = \begin{cases} 0 & \text{if } l \in \{0, \dots, 4\} \text{ and } r \in \{0, \dots, 4\} \\ 0.2 & \text{if } l \in \{0, \dots, 4\} \text{ and } r \in \{5, \dots, 9\} \\ 0.1 & \text{if } l \in \{5, \dots, 9\} \end{cases} \quad (2.5)$$

To generate one example we first sample the left digit using  $P(l)$  then the right using  $P(r|l)$ . As shown in Appendix A.1.1, we can compute the conditional entropies of one digit given the other to get  $H(l|r) = 2.0868$  and  $H(r|l) = 1.9360$ . Since  $H(l|r) > H(r|l)$ , the left digit is more informative of the right than the right is of the left. There is no cross-digit signal at test time, so features for the right and left digits should be completely decorrelated to generalize, but learned features will have some correlation between left and right. Intuitively, DeCov should help generalization in this scenario. Our experiments support this.

We use Caffe’s [Jia13] reference version of LeNet [LeC+95]. It has two convolution layers, each followed by pooling, then a fully connected layer with 500 hidden units which are shared between the two softmax layers. We apply DeCov and/or Dropout to the 500 hidden units of the fully connected layer.

**Tab. 2.1.: MNIST side by side results.** As expected, biasing right digits at train time so that they are weakly informed by left digits leads to lower performance on an unbiased test set. More importantly, DeCov provides greater improvements over the baselines on the right, confirming that it leads to better features when decorrelation is extremely likely to improve performance.

DeCov	Dropout	Left Digit			Right Digit		
		train	test	train - test	train	test	train - test
no	no	99.98 ± 0.01	97.94 ± 0.18	2.05 ± 0.19	100.00 ± 0.00	96.75 ± 0.24	3.25 ± 0.24
no	yes	99.99 ± 0.00	98.45 ± 0.04	1.54 ± 0.04	99.99 ± 0.00	97.39 ± 0.20	2.61 ± 0.20
yes	yes	99.97 ± 0.01	98.59 ± 0.12	1.38 ± 0.12	99.99 ± 0.00	97.81 ± 0.07	2.18 ± 0.06
yes	no	99.99 ± 0.00	<b>98.74 ± 0.03</b>	<b>1.25 ± 0.04</b>	99.99 ± 0.00	<b>97.99 ± 0.12</b>	<b>2.00 ± 0.12</b>
weight decay		99.97	97.86	2.11	99.97	96.21	3.76

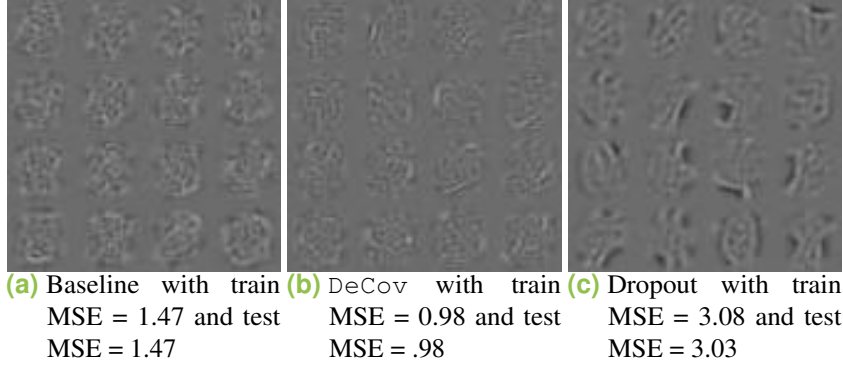
**Results.** (Tab. 2.1) reports the accuracy of left and right digit classifiers. Our injected dataset bias can be clearly seen in the lower test accuracy and higher train-test gap of the right classifier, indicating that all of our networks incorporate the train time bias into their predictions. We report mean accuracies across 4 trials, along with the standard deviation. We also compare the effect of Dropout.

The main result is that the gaps between the performance of DeCov and the baselines are larger for the biased right digit (*e.g.*, right digit test accuracy shows a  $\sim 0.6\%$  improvement when switching from Dropout-alone to DeCov-alone while the improvement for left digits is just  $\sim 0.3\%$ ). This suggests that the baselines pick up on the false bias and that DeCov does the best job of correcting for it. DeCov also improves generalization for both classifiers since test accuracy is higher in the bottom two rows and the train - test gap is lower in those rows. Combining Dropout with our DeCov loss hurts slightly, but we note that the error bars overlap in some cases, so this is not a statistically significant difference.

One skeptical hypothesis is that the DeCov loss is simply enforcing something akin to an L2 penalty on the weights. The experiments with DeCov and Dropout already use an L2 penalty, so this is unlikely, but a grid search over weights on this term shows it makes little difference. The best accuracies are reported in the last row of (Tab. 2.1).

## 2.4.2 MNIST Autoencoder

To offer a more qualitative point of comparison, we visualized learned features using the 2 layer autoencoder experiment from [Sri+14] (section 7). In this experiment an autoencoder is trained on raw pixels of single MNIST digits



**Fig. 2.3.:** Weights learned by the first layer of a 2 layer autoencoder are reshaped into images and visualized for a model with no DeCov or Dropout ((Fig. 2.3a)), a model with DeCov ((Fig. 2.3b)), and a model with Dropout ((Fig. 2.3c)).

using an encoder with 1 layer of 256 ReLU units and a decoder (untied weights) that produces 784 ( $28 \times 28$ ) ReLU outputs. (Fig. 2.3) shows the weights learned by the autoencoder (reshaped to align with the input image) and mean-square reconstruction errors.

Weight initialization turned out to be an important factor for the visualizations. Initializing all weights by sampling from  $U[-\sqrt{\frac{3}{n}}, \sqrt{\frac{3}{n}}]$  (based on [GB10]; as implemented in Caffe) led to visualizations as seen in [Sri+14] (the baseline looks like noise), but sampling weights from a Gaussian with mean 0 and standard deviation 0.001 led to baseline visualizations with faint digit outlines. The latter initialization was used in (Fig. 2.3).

One take-away is that MSE is significantly lower for DeCov than others. However, the key take-away is the qualitative difference between representations learned with Dropout and those learned with DeCov. Recall from Section 2.1 that Dropout reduces cross-covariance while DeCov explicitly minimizes it. Despite this intuitive similarity, the two lead to different learned representations.

## 2.4.3 Image Classification

### CIFAR10

CIFAR10 contains 60,000 32x32 images sorted into 10 distinct categories [KH09]. We training on the 50,000 given training examples and testing on the 10,000 specified test samples. Hyper-parameters (loss weights for DeCov and weight decay) are chosen by grid search on the standard train/val split.

We use Caffe’s quick CIFAR10 architecture, which has 3 convolutional layers followed by a fully connected layer with 64 hidden units and a softmax layer. The hidden fully connected layer is not followed by a non-linearity. The DeCov loss is added only to the 64 hidden units in the hidden fully connected layer. All reported results are average performance over 4 trials with the standard deviation indicated alongside.

**Tab. 2.2.:** CIFAR10 Classification. We can see that DeCov with Dropout leads to the highest test performance and the lowest train-test gap.

DeCov	Dropout	train	test	train - test
no	no	100.0 $\pm$ 0.00	75.24 $\pm$ 0.27	24.77 $\pm$ 0.27
no	yes	99.10 $\pm$ 0.17	77.45 $\pm$ 0.21	21.65 $\pm$ 0.22
yes	yes	87.78 $\pm$ 0.08	<b>79.75 <math>\pm</math> 0.17</b>	<b>8.04 <math>\pm</math> 0.16</b>
yes	no	88.78 $\pm$ 0.23	79.72 $\pm$ 0.14	9.06 $\pm$ 0.22
weight decay		100.0	75.29	24.71

**Results.** In Table 2.2, we again observe significant improvements when using the DeCov loss – there is a  $\sim 4.5\%$  improvement in test accuracy (over no regularization). Moreover, the DeCov loss reduces the gap between train and val accuracies by  $\sim 15\%$  (without Dropout) and  $\sim 16\%$  (with Dropout)!

Comparing the four combinations, we see that using DeCov alone provides a larger improvement than using Dropout. Using both DeCov and Dropout



further improves the generalization (as measured by the gap in train and test accuracies), but the improvement in absolute test performance does not seem statistically significant.

We again test if L2 weight decay can provide similar improvements and find once again that the best setting gives little improvement over the baseline.

One promise of regularization is the ability to train larger networks, so we increase the size of our CIFAR10 network. We add another fully connected layer to the network used in the previous experiment, double the number of filters in each convolutional layer, and double the number of units in the fully connected layers. This larger network performs better than the smaller version – all accuracies are higher than corresponding entries in (Tab. 2.2). However, there are the stronger indications of overfitting in this network – specifically, the train accuracies are much higher than test accuracies (when compared to the previous network). (Tab. 2.3) shows the results. We observe similar trends as the previous experiment – there are significant gains from using DeCov alone compared to Dropout alone, and there is a further slight improvement in combining both. Using Dropout alone gives a  $\sim 1.5\%$  boost in test accuracy, while using DeCov alone provides a  $\sim 4\%$  increase in test accuracy. Using both yields roughly the same test performance, but the trainval and test gap is further reduced.

**Tab. 2.3.:** CIFAR10 Classification with a bigger version of the base network

DeCov	Dropout	(train+val)	test	(train+val) - test
no	no	100.00	77.38	22.62
no	yes	100.00	79.93	20.07
yes	yes	96.76	<b>81.68</b>	<b>15.08</b>
yes	no	98.15	81.63	16.52

## CIFAR100

To scale up our experiments, we move to CIFAR100 [KH09]. We use the same architecture as the base architecture for CIFAR10 and hold out the last 10,000 of the 50,000 train examples for validation. Table 2.4 shows that Dropout alone highest higher test performance than DeCov alone, but DeCov leads to a smaller train-test gap. Using both regularizers not only achieves the highest test accuracy, but also the smallest train-test gap ( $\sim 34\%$  smaller than using neither regularizer). This suggests that the two regularizers may have complementary effects.

**Tab. 2.4.:** CIFAR100 Classification Accuracies

DeCov	Dropout	train	test	train - test
no	no	99.77	38.52	61.25
no	yes	87.35	43.55	43.80
yes	yes	72.53	<b>45.10</b>	<b>27.43</b>
yes	no	77.92	40.34	37.58

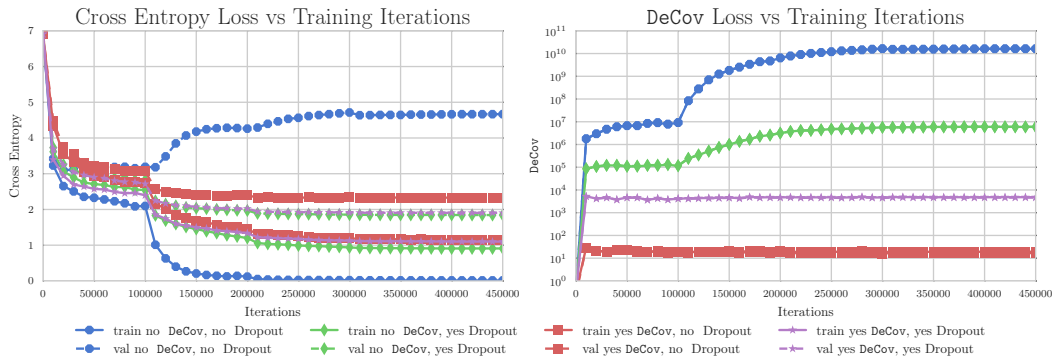
One more problem comes with the question of how to weight the DeCov loss. All of our experiments use grid search to pick this hyper-parameter. The optimal weight varies across datasets, but we have found consistency across variations in architecture. We varied both the DeCov weight and the number of hidden units in the fully connected layer to which DeCov is applied, training a new network for each setting. The best DeCov weight (0.1) is consistent for a range of hidden activation sizes in this dataset, though it is different in other experiments.

## ImageNet

Now we explore results for networks trained for ImageNet classification, starting by applying DeCov to fc6 and fc7 in AlexNet [KSH12]. The last 50,000 of the ILSVRC 2012 train images are held out for validation. Our implementation

comes from Caffe. In particular, it uses a fixed schedule that multiplies the learning rate by 1/10 every 100,000 iterations (see jumps in (Fig. 2.4)). We do not use early stopping and do not perform color augmentation.

In (Fig. 2.4) we notice that when neither of the two regularizers – Dropout or DeCov– are applied (blue line), the network overfits (it even gets 100% train accuracy), and the DeCov loss (hidden activation redundancy) is higher than with any other combination of the regularizers. Applying either of the regularizers also causes a synchronous drop in both losses. Explicitly minimizing the DeCov loss naturally leads to much lower DeCov losses, and we notice that this coincides with significantly reduced overfitting. Interestingly, Dropout results in relatively lower DeCov loss too, even when DeCov is not optimized for. This is further indication of the link between redundant activations and overfitting.

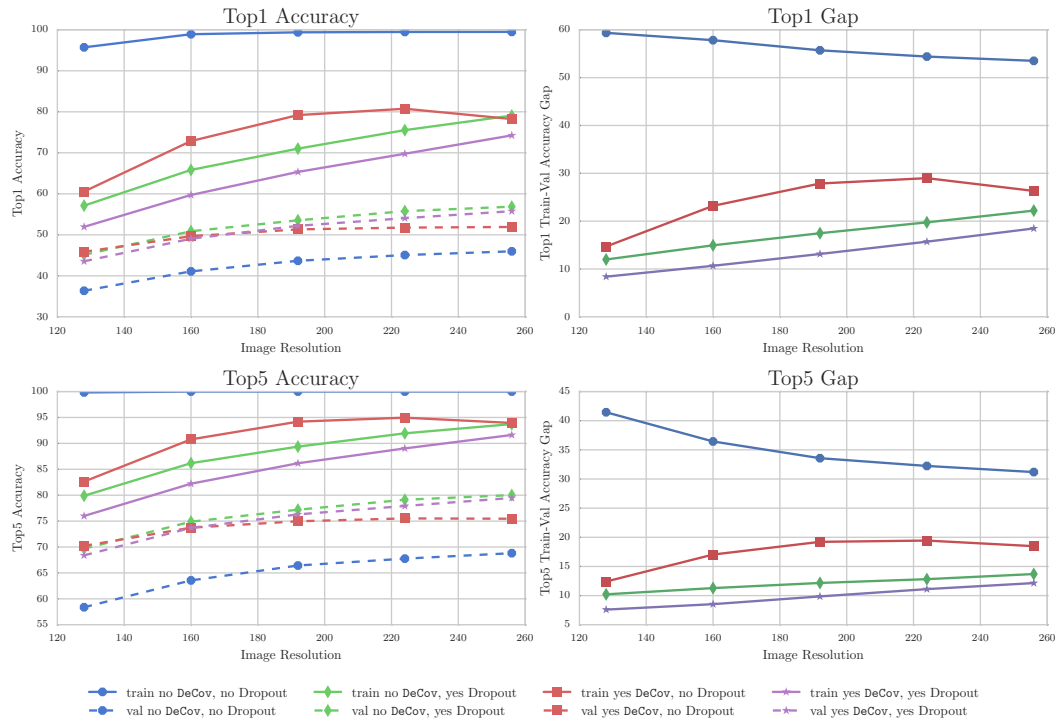


**Fig. 2.4.:** Cross Entropy and DeCov losses over the course of training AlexNet with 256x256 images. Note that the DeCov val curves are hidden by the train curves. Interestingly, DeCov is reduced even by Dropout, though not nearly as much as when it is explicitly minimized.

(Fig. 2.5) shows accuracies across different image resolutions we used to train AlexNet. AlexNet is typically trained with 256x256 images, but training with smaller images is faster <sup>1</sup> and reduces the number of parameters in the network. Smaller images (we use 128x128, 160x160, 192x192, and 224x224) lead to smaller feature maps output by pool5, so the dense connection between pool5

<sup>1</sup>Using CuDNNv3, AlexNet with 128x128 inputs takes 103ms averaged over 50 runs to compute a forward and backward pass. For 256x256 images this time is 449ms.

and fc6 has fewer parameters, the model has less capacity, and it's less likely to overfit. For example, images scaled to 256x256 (taking 227x227 crops<sup>2</sup>) lead to a weight matrix with 38 million parameters while 128x128 images (with 99x99 crops) result in a 4 million parameter matrix. Generally, accuracies (left plots) and the train-val gap (right plots) have a slight positive slope, confirming that performance and overfitting increase with resolution and model capacity. Note that the DeCov loss weight was tuned using grid search at each resolution both with and without Dropout.



**Fig. 2.5.:** ImageNet classification performance using AlexNet. Plots on the left show training and validation (ILSVRC 2012 validation set) accuracy at different resolutions. Note how all curves have a much lower train-val gap than the (blue) baseline.

We see that Dropout alone (green) usually has the best val accuracy, which is slightly higher than the two losses combined (purple) and a couple points higher than DeCov alone (red) at higher resolutions. At the lowest resolution Dropout alone is tied with DeCov alone. Dropout also reduces overfitting more than

<sup>2</sup>At train time crops are sampled and mirrored randomly. At test time only the center 227x227 crop is used.

DeCov, though both independently reduce overfitting by a large margin – from 59.35% to 14.7% in the case of DeCov @ 128x128.

Finally, we test our new regularizer on ILSVRC 2012 with one more architecture – the Network in Network [LCY13].<sup>3</sup> This architecture is fully convolutional: it contains 4 convolutional layers, with 96, 256, 384, and 1024 feature maps, respectively. Between each of these layers and after the last are two convolutional layers which have 1x1 kernels, which further process each feature map output by the main convolutional layers before being fed into the next layer. To produce 1000 softmax activations, 1000 feature maps are averaged over spatial locations to produce one feature vector. We applied DeCov to these average pooled feature vectors.

Interestingly, this architecture has much less overfitting than AlexNet. However, adding a DeCov loss still decreases overfitting substantially and improves validation accuracy. There is a small boost in performance on validation accuracies and a significant decrease of  $\sim 3\%$  (for top 1) and  $\sim 2\%$  (for top 5) in the train - val gap.

**Tab. 2.5.:** ImageNet Classification Accuracies with Network in Network

DeCov	Dropout	ILSVRC 2012 train top 1	ILSVRC 2012 val top 1	train - val
no	no	71.68	58.67	13.01
no	yes	71.32	58.95	12.37
yes	yes	68.28	<b>59.08</b>	<b>9.20</b>
yes	no	68.33	58.85	9.48
DeCov	Dropout	ILSVRC 2012 train top 5	ILSVRC 2012 val top 5	train - val
no	no	89.91	81.18	8.73
no	yes	89.63	81.53	8.10
yes	yes	87.99	<b>81.94</b>	<b>6.05</b>
yes	no	87.88	81.57	<b>6.05</b>

<sup>3</sup>This is the model provided in the Caffe Model Zoo: <https://gist.github.com/mavenlin/d802a5849de39225bcc6>

## 2.5 Discussion and Conclusion

**Fine-tuning.** In the experiments we presented, networks were always trained from scratch, but we also tried fine-tuning networks in different scenarios. During our ImageNet experiments we fine-tuned both the Network in Network and AlexNet architectures initialized with parameters that weren't trained with a DeCov loss, but were trained with Dropout. In both cases performance either stayed where it was at fine-tuning initialization or it decreased slightly (within statistical significance). We found similar results when fine-tuning for other tasks like attribute classification (fine-tuning AlexNet) and object detection (Fast RCNN [Gir15]).

This, along with some cases where combining Dropout and DeCov decreases performance slightly suggest that the DeCov loss may possibly be acting adversarially to activations learned by Dropout. Fine-tuning with DeCov is an interesting direction for future work.

**Trends.** All of our experiments strongly indicate two clear trends:

1. DeCov reduces overfitting as measured by the gap between train and test performance.
2. DeCov acts as a regularizer: performance with DeCov is always better than performance without either DeCov or Dropout.

To be clear, the results do not support that Dropout can be completely replaced by DeCov, but simply that in a number of scenarios DeCov is a useful alternative and their combination almost always works the best. Our loss clearly has desirable regularization properties at the expense of one extra hyper-parameter to tune.

In this work, we proposed a new `DeCov` loss which explicitly penalizes the covariance between the activations in the same layer of a neural network in an unsupervised fashion. This loss acts as a strong regularizer for deep neural networks, where overfitting is a major problem and Dropout has been required to get large models to generalize well. We show that `DeCov` competes well against Dropout over a range of experiments which investigate different scales, datasets and architectures.

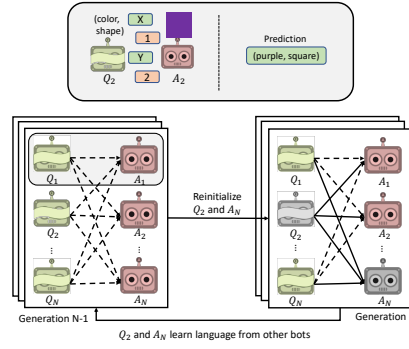
# Emergence of Compositional Language with Deep Generational Transmission

## 3.1 Introduction

Compositionality is an important structure of language that reflects a disentangled understanding of the world – enabling the expression of infinitely many concepts using finitely many elements. Agents that have compositional understandings of the world generalize in obviously correct ways even in the face of limited training examples [LB18]. For example, an agent with a compositional understanding of `blue squares` and `purple triangles` should also understand `purple squares` without directly observing any of them. Developing artificial agents that can ground, understand, and produce compositional (and therefore more interpretable) language could greatly improve generalization to new instances and ease human-AI interactions.

In building theories of how compositionality emerges in human languages, work in evolutionary linguistics looks to the process of cultural transmission [Kir01; KCS08]. Cultural transmission of language occurs when a group of agents pass their language on to a new group of agents, *e.g.* parents who teach their children





**Fig. 3.1.:** We introduce cultural transmission into language emergence between neural agents. Start with the goal-oriented dialog task at the top of the figure (similar to that of Kottur et al. [Kot+17]). During learning we periodically replace some agents with new ones (gray agents). These new agents do not know any language, but instead of creating one they learn it from older agents. This creates generations of language that become more compositional over time.

to speak as they do. Because this education is incomplete and biased, it allows the language itself to change over time via a process known as *cultural evolution*. This paradigm [KGS14] explains the emergence of compositionality as a result of expressivity and compressibility – *i.e.* to be most effective, a language should be expressive enough to differentiate between all possible meanings (*e.g.*, objects) and compressible enough to be learned easily. Work in the evolutionary linguistics community has shown that over multiple ‘generations’ these competing pressures result in the emergence of compositional languages both in simulation [Kir01] and with human subjects [KCS08]. These studies aim to understand humans whereas we want to understand and design artificial neural networks.

Approaching the problem from another direction, recent work in AI has studied language emergence in such multi-agent, goal-driven tasks. These works have demonstrated that agent languages will emerge to enable coordination-centric tasks to be solved without direct or even indirect language supervision [Foe+16; SSF16; LPB17; Das+17a]. However, the resulting languages are usually not compositional and are difficult to interpret, even by other machines [ADK17].

Some existing work has studied means to encourage compositional language formation [MA18; Kot+17], but these settings study fixed populations of agents – *i.e.* examining language within a single generation.

**In this work we bridge these two areas – examining the effect of generational cultural transmission on the compositionality of emergent languages in a multi-agent, goal-driven setting.**

We study this in the context of a cooperative dialog-based reference game involving two agents communicating in discrete symbols [Kot+17]; an example dialog is shown at the top of (Fig. 3.1). To examine cultural transmission, we extend this setting to a population of agents (bottom of (Fig. 3.1)) and introduce a simple mechanism to induce the expressivity and compressibility pressures inherent in cultural transmission. Specifically, we periodically re-initialize some subset of the agents in the population. In order to perform well at the task, the population’s emergent language must be sufficiently expressive to reference all the objects (expressivity) and must be easily learnable by these ‘new’ agents (compressibility). The new agents have a randomized language whereas the surviving agents already know a grounded language. This “knowledge gap” creates an implicit ‘teaching’ setting that is analogous to the explicit transmission stage in models of *iterative learning* [Kir01].

Through our experiments and analysis, we show that periodic agent replacement is an effective way to induce cultural transmission and yields more compositionally generalizable language in our setting. To summarize, our contributions are:

- We propose a method for inducing implicit cultural transmission in neural language models.

- We measure the similarity between agent languages and verify cultural transmission has occurred as a result of our periodic agent replacement protocol.
- We show our cultural transmission procedure induces compositionality in neural language models, going from 13% accuracy on a compositionally novel test set to 46% in the best configuration. Further, we show this is complementary with previous priors which encourage compositionality.

## 3.2 Task & Talk: A Testbed for Compositional Language

### Emergence

We consider the cooperative *Task & Talk* reference game introduced in [Kot+17]. Shown in the top of (Fig. 3.1), the game is played by two agents – one who observes an attributed object – *e.g.* (`purple`, `solid`, `square`) – and another who is given a task to retrieve a subset of these attributes over the course of the dialog – *e.g.* (`color`, `shape`). The dialog itself consists of two rounds of agents exchanging single-token utterances from fixed vocabularies. At the end of the dialog, the task-aware agent must report the requested attributes and both agents are rewarded for correct predictions. This causes a language grounded in the objects to *emerge* because there is no other way to solve the task.

A compositional solution to this task can look like a question-answer style dialog where the task-aware agent queries the other for specific attributes (top of (Fig. 3.1)) – *e.g.* uttering “X” requesting the `color` to which the other agent replies “1” indicating `purple`. Importantly, this pattern would persist regardless of the other attribute values of the object (*e.g.* for all (`purple`, `*`, `*`) objects). However, as there is no grounding supervision provided, agents must learn to associate specific meanings to specific words and it is unlikely

for compositional languages to emerge purely by chance. Given the same color task, an non-compositional agent might use “1” for (purple, solid, square) and then “2” for a novel instance (purple, solid, circle). Other agents have no way to know that “2” means purple instead of “1”, so compositional language is essential for generalization to compositionally novel instances.

**Models..** To formalize this setting, let Q-bot and A-bot be agent policies parameterized by neural networks  $Q$  and  $A$  respectively. At each round  $t$ , Q-bot observes the task  $x_Q$  and it’s memory of the dialog so far  $h_Q^{t-1}$  and produces a single-token utterance  $m_Q^t \in \mathcal{V}$  from the vocabulary  $\mathcal{V}$ . Functionally,  $m_Q^t, h_Q^t = Q(m_A^{t-1}, x_Q, h_Q^{t-1})$  where  $m_A^{t-1}$  is A-bot’s reply in the previous round. Likewise, A-bot responds by computing  $m_A^t, h_A^t = A(m_Q^t, x_A, h_A^{t-1})$  where  $x_A$  is the object instance represented symbolically by concatenating 3 one-hot vectors, one per attribute. After two rounds, Q-bot must respond to the task, predicting the requested attribute pair  $\hat{u} = U(x_Q, h_Q^T)$  as a function of the task and Q-bot’s final memory state. Both agents are rewarded if both attributes are correct (no partial credit). We follow the neural network architectures of  $Q$ ,  $A$ , and  $U$  from [Kot+17].

**Measuring Compositional Generalization..** Kottur et al. [Kot+17] generated a synthetic dataset consisting of three attribute types (color, shape, style) each with four values (e.g., red, blue, square, star, dotted, solid, ...) and six tasks, one task for each ordered pair of different attribute types. This results in 64 unique instances and 384 task-instance pairs. To evaluate compositionality, Kottur et al. [Kot+17] held out 12 random instances for testing. Given the closed-world set of instances, these 12 triplets of attributes is not seen during training; however, each individual value is seen in other triplets that do appear in training. As such, test accuracy is a measure of compositional generalization.

**Shortcomings of [Kot+17] Evaluation..** In our investigations, we found some shortcomings in the evaluation protocol of [Kot+17]. First, the authors do not report variance over multiple runs or different random test-sets which we found to be significant. Second, the strategy of randomly selecting the test set can still reward some only partially compositional strategies. For instance, suppose agents develop a language that uses single words to refer to attribute pairs like (red, \*, triangle) and (red, filled, \*). Such agents might generalize to an unseen instance (red, filled, triangle) by composing the ‘paired’ words above instead of disentangling individual attributes.

We make two modifications to address these issues. Our results are reported as means and variances estimated from multiple training runs with 4 different random seeds and 4-way cross-validation (16 experiments). We also introduce a harder dataset where instead of withholding random individual instances (e.g., (green, dotted, triangle), ...) as in [Kot+17], we withhold all instances for a set of attribute pairs (e.g., (green, dotted, \*), (red, solid, \*), ...). We will refer to datasets generated in this fashion as **novel pair** and the original dataset as **novel instance**. We report on both settings for comparison (see appendix A.1), but find our new setting to be significantly more challenging in practice – requiring a stricter notion of compositionality more closely aligned with human intuitions about these attributes.

### 3.3 Compositional Language Emergence with Cultural

#### Transmission

In iterative learning models of cultural transmission from evolutionary linguistics, competing pressures towards expressivity and compressibility have been

shown to induce compositionality over multiple ‘generations’ of language transfer [Kir01; KCS08]. The goal-driven nature of our reference game already encourages expressivity – agents must be able to refer to the objects in order to succeed. To introduce compressibility pressure and parallel literature in evolutionary linguistics, we introduce a population of agents which regularly has members replaced by new agents that lack any understanding of the remaining population’s language. As this paradigm lacks explicit teaching steps where new agents are trained to ground existing words, we consider this approach as a means of implicit cultural transmission.

---

**Algorithm 1:** Training with **Replacement** and **Multiple Agents**

---

```

1 for epoch  $e = 1, \dots, N_{\text{epochs}}$  do
2   Sample Q-bot  $i_Q$  from  $\mathcal{U}\{1, N_Q\}$  and A-bot  $i_A$  from  $\mathcal{U}\{1, N_A\}$ 
3   for  $x_Q, x_A, u$  in each batch do
4     for dialog rounds  $t = 1, \dots, T$  do
5        $m_Q^t, h_Q^t = Q^{i_Q}(m_A^{t-1}, x_Q, h_Q^{t-1})$ 
6        $m_A^t, h_A^t = A^{i_A}(m_Q^{t-1}, x_A, h_A^{t-1})$ 
7        $\hat{u} = U^{i_Q}(x_Q, h_Q^T)$ 
8       Policy gradient update w.r.t. both Q-bot and A-bot parameters
9   if  $e \bmod E = 0$  then
10    Sample replacement set  $B$  under policy  $\pi$  and re-initialize all agents
11    in  $B$ 
11 return all Q-bots and A-bots.

```

---

**Populations of Agents..** We consider a population of Q-bots  $\{Q^1, \dots, Q^{N_Q}\}$  and a population of A-bots  $\{A^1, \dots, A^{N_A}\}$  with each agent having a different set of parameters. At each iteration during learning, we sample a random Q-bot-A-botpair to interact and receive updates – *i.e.* the red line (2) in Algorithm 1. As any Q-bot may be made to communicate with any A-bot, there is pressure for the population to adopt a unified language. Likewise, when an agent is reinitialized it will receive positive reward much more quickly when it happens to use language that its conversational partners understand. Furthermore, ‘compressible’ languages that are easier to learn will result in greater reward for the population in the face of periodic re-initialization of agents.

Introducing multiple agents may in itself add compressibility pressure and improve generalizations even without replacement [RMLA18]. Agents in a population have to model minor linguistic differences between conversational partners given the same memory capacity. Further, each agent provides another potential language variation that can be mimicked and perpetuated—increasing language diversity early in training. We examine these effects through no-replacement baselines, but find that generational pressure where some agents know less than others can also be important for compositionality in our setting.

**Replacement.** In order to create a notion of ‘generations’ we replace agents periodically. Let  $\pi$  be a replacement strategy, returning a subset of the population. Every  $E$  epochs, we call  $\pi$  and reinitialize the parameters and optimizers for the returned agents (blue lines 9-10 in Algorithm 1). We investigate three settings of  $\pi$  (see appendix A.2 for more details):

- **Uniform Random.** Sample an A-bot and Q-bot from uniform random distributions.
- **Epsilon Greedy.** With probability  $1 - \varepsilon$  replace the A-bot and Q-bot with the lowest validation accuracy. We use  $\varepsilon = 0.2$  in our experiments.
- **Oldest.** Replace the oldest A-bot and Q-bot, breaking ties with uniform random sampling.

## 3.4 Experimental Setting

**Experimental Setting.** We evaluate on both our novel pair dataset and the novel instance dataset from Kottur et al. [Kot+17] (see appendix A.1), as described in Section 3.2. All results are reported as means and variances computed from a total of 16 trials (four random seeds each with 4-way cross-validation).

We report accuracy based on Q-bot getting both elements of the task correct – corresponding to the more restrictive “Both” setting from [Kot+17].

Kottur et al. [Kot+17] examined a series of increasingly restrictive settings in order to study conditions under which compositionality emerges. The primary variables are whether A-bot has memory (ablated by setting  $h_A^t=0$ ) and the vocabulary sizes  $\mathcal{V}_Q$  and  $\mathcal{V}_A$  for Q-bot and A-bot respectively. For comparison we also evaluate in these settings: **Minimal Vocab** ( $V_Q=3, V_A=4$ ). **Memoryless + Minimal Vocab** ( $V_Q=3, V_A=4, h_A^t=0$ ), **Overcomplete** ( $V_Q=V_A=64$ ). We also introduce **Memoryless + Overcomplete** ( $V_Q=V_A=64, h_A^t=0$ ) to complete the cross product of settings and examine the role of memory restriction in overcomplete vocabularies.

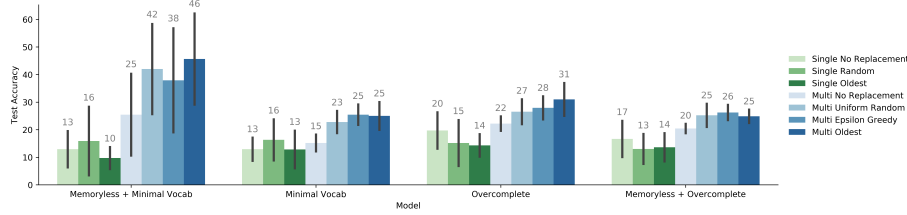
The Memoryless + Minimal Vocabulary setting results in the best compositional generalization; however, this is an extreme setting – requiring not only that the minimum number of groundable symbols be known but also that A-bot not be able to remember its previous utterance. While we do report these settings and see quite large performance gains due to cultural transmission, we are mainly interested in the more realistic Overcomplete setting where a large pool of possible tokens is provided and both dialog agents have memory.

**Model and Training Details.** Our A-bots and Q-bots have the same architecture as in Kottur et al. [Kot+17]. All agents are trained with  $E = 25000$ , a batch size of 1000,<sup>1</sup> and the Adam [KB15] optimizer (one per bot) with learning rate 0.01. In the Multi Agent setting we use  $N_A = N_Q = 5$ . We stop training after 8 generations (199000 epochs Multi Agent; 39000 epochs Single Agent). This differs from Kottur et al. [Kot+17], which stopped once train accuracy reached 100%. Further, we do not mine negatives.

---

<sup>1</sup>All 384 instances (64 objects  $\times$  6 tasks) fit in 1 batch.





**Fig. 3.2.:** Test set accuracies (with standard deviations) are reported against our new harder dataset using models similar to those in [Kot+17]. Our variations on cultural transmission (darker blue bars) outperform the baselines without cultural transmission.

**Baselines.** These help isolate the effects of our approach.

- **Single Agent Populations.** We ablate the effect of multi-agent populations by training individual A-bot-Q-botpairs (*i.e.* populations with  $N_A = N_Q = 1$ ). We apply the *uniform random* (either A-botor Q-botat random) and *oldest* (alternating between A-botand Q-bot) replacement strategies to these agents; however, the *epsilon greedy* strategy is not well-defined here. In this setting we decrease  $E$  from 25000 to 5000 to keep the average number of gradient updates for each agent constant with respect to the multi-agent experiments.
- **No Replacement.** We also consider the effect of replacing no agents at all, but still allowing the agents to train for the full 199,000 (39,000) epochs. Improvement over this baseline shows the gains from our replacement strategy under identical computational budgets.

## 3.5 Results and Analysis

### 3.5.1 Impact of Cultural Transmission on Compositional Generalization

Results with standard deviations against our harder dataset are reported in (Fig. 3.2). We compared methods and models using dependent paired t-tests

and reported the resulting p-values in Section A.4 of the appendix. Result on the original Task & Talk dataset are in Section A.1 of the appendix.

**Cultural transmission induces compositionality.** Our main result is that cultural transmission approaches outperform baselines without cultural transmission. This can be seen by noting that for each model type in (Fig. 3.2), the 3 darker blue bars (Multi Agent Replacement approaches) are largest. After running a dependent paired t-test against all pairs of baselines and cultural transmission approaches we find a meaningful difference in all cases ( $p \leq 0.05$ ). This is strong support for our claim that our version of cultural transmission encourages compositional language because it causes better generalization to novel compositions of attributes.

Next we go on to discuss some additional trends we hope the community will find useful.

**Population dynamics without replacement usually lead to some compositionality.** The *Multi Agent No Replacement* policies usually outperform than the *Single Agent No Replacement* policies, though the difference isn't very significant in the except in the *Overcomplete* and *Minimal Vocab* settings. This agrees with recent work from evolutionary linguistics, where multiple agents can lead to compositionality without generational transmission [RMLA18].

**Variations in replacement strategy tend to not affect performance.** The *Multi Agent Uniform Random/Epsilon Greedy/Oldest* replacement strategies are not largely or consistently different from one another across model variations. This suggests that while some agent replacement needs to occur, it is not critical whether agents with worse language are replaced or whether there is a pool of similarly typed agents to remember knowledge lost from older generations. The

main factor is that new agents learn in the presence of others who already know a language.

**Cultural transmission is complementary with other factors that encourage compositionality..** As in Kottur et al. [Kot+17], we find the *Memoryless* + *Small Vocab* model is clearly the best. This agrees with factors noted elsewhere [Kot+17; MA18; NPJ00] and shows how many different factors can affect the emergence of compositionality.

**Removing memory makes only minor differences..** Removing memory makes no difference (negative or positive) in Single Agent settings, but it can have a relatively small effect in Multi Agent settings, helping *Small Vocab* models and hurting *Overcomplete* models. While our approach is complementary with minimizing vocab size to increase compositionality, its makes memory removal less useful. As the *Memoryless* + *Overcomplete* setting has not been reported before, these results suggest that the relationship between inter-round memory and compositionality is not clear.

Overall, these results show that adding cultural transmission to neural dialog agents improves the compositional generalization of the languages learned by those agents in a way complementary to other priors. It thereby shows how to transfer the cultural transmission principle from evolutionary linguistics to deep learning.

### 3.5.2 Is Generational Transmission Occurring?

Because it is implicit, cultural transmission may not actually be occurring; improvements may be from other sources. How can we measure cultural transmission? We focus on A-bots and take a simple approach. We assume

that if two A-bots ‘speak the same language’ then that language was culturally transmitted. There is a combinatorial explosion of possible languages that could refer to all the objects of interest, so if the words that refer to the same object for two agents are the same then they were very likely transmitted from the other agents, rather than similar languages emerging from scratch just by chance. This leads to a simple approach: consider pairs of bots and see if they say similar things in the same context. If they do, then their language was likely transmitted.

More formally, consider the distribution of tokens A-bot  $A^i$  might use to describe its object  $x_A$  when talking to Q-bot  $Q^k$ :  $p_{k,i}(m_A^t|x_A)$  or  $p_{k,i}$  for short. We want to know how similar  $A^i$ ’s language is to that of another A-bot  $A^j$ . We’ll start by comparing those two distributions by computing the KL divergence between them and then taking an average over context (objects, Q-bots, and dialog rounds) to get our pairwise agent language similarity metric  $D_{ij}$ :

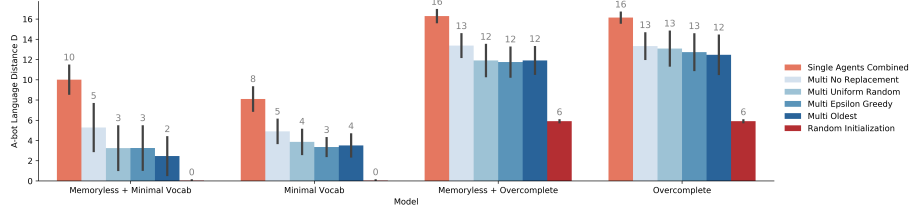
$$D_{ij} = \hat{E}_{x_A, k, t} \left[ D_{KL} \left( p_{k,i}(m_A^t|x_A), p_{k,j}(m_A^t|x_A) \right) \right] \quad (3.1)$$

Taking another average, this time over all pairs of bots (and also random seeds and cross-val folds), gives our final measure of language similarity reported in (Fig. 3.3).

$$D = \hat{E}_{i,j \text{ s.t. } i \neq j} [D_{ij}] \quad (3.2)$$

$D$  is smaller the more similar language is between bots. Note that even though  $D_{ij}$  is not symmetric (because KL divergence is not),  $D$  is symmetric because it averages over both directions of pairs.

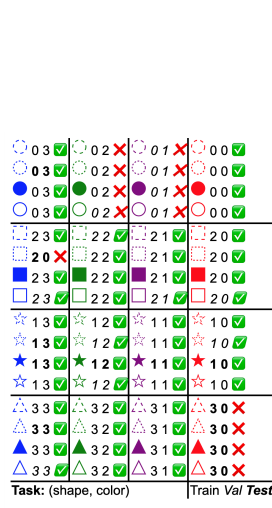
We compute  $D$  by sampling an empirical distribution over all messages and observations, taking 10 sample dialogues in each possible test state  $(x_A, x_Q)$  of



**Fig. 3.3.:** Do bots in a population learn similar languages? On the y-axis (eq. (3.2)) lower values indicate more similar language. Bots from our method speak similar languages, but independently evolved agents do not. Thus our implicit procedure induces cultural transmission.

the world using the final populations of agents as in (Fig. 3.2). Note that this metric applies to a group of agents, so we measure it for only the *Multi Agent* settings, including two new baselines colored red in (Fig. 3.3). The *Single Agents Combined* baseline trains 4 *Single Agent No Replacement* models independently then puts them together and computes  $D$  for that group. These agents only speak similar languages by chance, so  $D$  is high. The *Random Initialization* baseline evaluates language similarity using newly initialized models. These agents have about a uniform distribution over words at every utterance, so their languages are both very similar and useless. For each model these baselines act like practical (not strict) upper and lower bounds on  $D$ , respectively.

(Fig. 3.3) shows this language dissimilarity metric for all our settings. As we expect, the paired *Single Agents* are highly dissimilar compared to agents from *Multi Agent* populations. Further, all the replacement strategies result in increased language similarity—although the degree of this effect seems dependent on vocabulary setting. This provides some evidence that cultural transmission is occurring in *Multi Agent* settings and is encouraged by the replacement strategy in our approach. While all *Multi Agent* settings resulted in language transmission, our replacement strategies results in more compositional languages due to repeated teaching of new generations of agents.



**Fig. 3.4.:** All conversations between Q-bot 4 and Q-bot 3 for the `(shape, color)` task. These bots were trained in the *Multi Agent Oldest* setting. The figure shows A-bot’s utterances for each object and whether or not Q-bot guessed the object correctly, as described in Section 3.5.3. This language is compositional because each token refers to a color or shape.

### 3.5.3 Visualizing Emergent Languages

In this section we visualize the language learned by a pair of bots to show its compositionality. In the appendix we compare these bots to others at different stages of learning and from earlier generations to help understand how the language developed over generations.

Figure 3.4 shows all 64 conversations between Q-bot 4 and A-bot 3 for the `(shape, color)` task. These bots are from the 8th generation of the *Multi Agent Oldest* setting.

To interpret the visualization, start by looking at only the dashed blue circle in the top left. To the right of it are the two tokens “0” and “3”, which are the words A-bot used to describe the object in the two dialog rounds. The green check one more step to the right indicates that Q-bot was able to guess the `circle blue` from these tokens. Now look at all 4 blue circles in the top left grid cells. Only `shape` and `color` matter for this task, so A-bot uttered “0 3” for every blue circle, appropriately ignoring style (*i.e.*, dashed, dotted, filled, or solid).

By looking at the entire visualization with its 4x4 grid delineated by black separators we can see that the language is indeed compositional. Rows of the 4x4 grid group objects by shape and columns group objects by color. This makes it convenient to qualitatively evaluate language compositionality with respect to the (`shape`, `color`) task. If A-bot’s language is compositional then it should use one token to indicate row / shape and one token for column / color.

Looking at the first row, A-bot’s first utterance is always “0”, but is not “0” anywhere else, so when “0” is uttered first it means `circle`. Similarly, A-bot’s second utterance is always “3” in the first column, so “3” means `blue`. Continuing with this analysis we find each character has meaning: (0=`circle`, 2=`square`, 1=`star`, 3=`triangle`), and (3=`blue`, 2=`green`, 1=`purple`, 0=`red`). Individual symbols have meaning, so the language is compositional.

## 3.6 Related work

**Language Evolution Causes Structure.** Researchers have spent decades studying how unique properties of human language like compositionality could have emerged. There is general agreement that people acquire language using a combination of innate cognitive capacity and learning from other language speakers (cultural transmission), with the degree of each being widely disputed [Per02; PB90]. Both innate cognitive capacity and specific modern human languages like English co-evolved [Bri00] via biological [PB90] and cultural [Tom99; Smi06] evolution, respectively.

In particular, explanations of how the cultural evolution of languages could cause structure like compositionality are in abundance [NK99; NPJ00; SKB03; Bri02; Vog05; KGS14; Spi+17]. An important piece of the explanation of linguistic

structure is the iterated learning model [KGS14; Kir01; KCS08] used to motivate our approach. Indeed it shows that cultural transmission causes structure in computational [Kir01; Kir02; CK03; SKB03] and human [KCS08; CTK09; SPK10] experiments. Even though cultural transmission may aid the emergence of compositionality, recent results in evolutionary linguistics [RMLA18] and deep learning [Kot+17; MA18] also emphasize other factors.

While existing work in deep learning has focused on biases that encourage compositionality, it has not considered settings where language is permitted to evolve over generations of agents. We have shown such an approach is viable and even complementary with other approaches.

**Language Emergence in Deep Learning.** Recent work in deep learning has increasingly focused on multi-agent environments where deep agents learn to accomplish goals (possibly cooperative or competitive) by interacting appropriately with the environment and each other. Some of this work has shown that deep agents will develop their own language where none exists initially if driven by a task which requires communication [Foe+16; SSF16; LPB17]. Most relevant is work which focuses on conditions under which *compositional* language emerges as deep agents learn to cooperate [MA18; Kot+17]. Both Mordatch and Abbeel [MA18] and Kottur et al. [Kot+17] find that limiting the vocabulary size so that there aren't too many more words than there are objects to refer to encourages compositionality, which follows earlier results in evolutionary linguistics [NPJ00]. Follow up work has continued to investigate the emergence of compositional language among neural agents, mainly focusing on perceptual as opposed to symbolic input and how the structure of the input relates to the tendency for compositional language to emerge [CLF18; HT17; Laz+18]. Other work has shown that Multi Agent interaction leads to better emergent translation [Lee+18], but it does not measure compositionality.



**Cultural Evolution and Neural Nets.** Somewhat recently, Bengio [Ben12] suggested that culturally transmitted ideas may help in escaping from local minima. Experiments in Gülçehre and Bengio [GB16] support this idea by showing that supervision of intermediate representations allows a more complex toy task to be learned. Unlike our work, these experiments use direct supervision provided by the designed environment rather than indirect and implicit supervision provided by other agents.

Two concurrent works examine the role of periodic agent replacement on language emergence – albeit in different environments. In Li and Bowling [LB19] replacement is used to encourage languages to be easy to teach, and this in turn causes compositionality. In Dagan et al. [DHB19] neural language is transmitted through a bottleneck caused by replacement. The resulting language has increased efficiency and effectiveness, with further results showing that co-evolving the agents themselves with the language amplifies the effect. Both of these works support our central observations.

## 3.7 Conclusion

In this work we investigated cultural transmission in deep neural dialog agents, applying it to language emergence. The evolutionary linguistics community has long used cultural transmission to explain how compositional languages could have emerged. The deep learning community, having recently become interested in language emergence, has not investigated that link until now. Instead of explicit models of cultural transmission familiar in evolutionary linguistics, we favor an implicit model where language is transmitted from generation to generation only because it helps agents achieve their goals. We show that this does indeed cause cultural transmission and compositionality.

**Future work.** While our work used an implicit version of cultural transmission, we are interested in the effect of explicit versions of cultural transmission on language structure. Cultural transmission may also provide an appropriate prior for neural representations of non-language information.

Dialog Without Dialog:

Learning Image

Discriminative Dialog

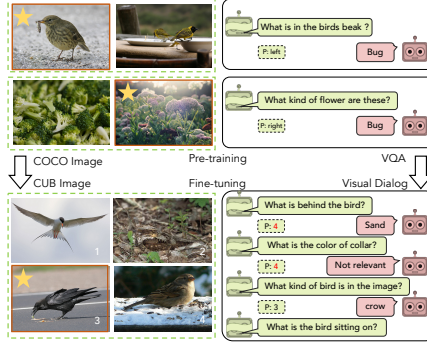
Policies from Single Shot

Question Answering Data

## 4.1 Introduction

One goal of AI is to enable humans and computers to communicate naturally with each other in grounded language to achieve a collaborative objective. Recently the community has studied this in the context of goal oriented dialog, where agents need to talk to perform tasks like booking a flight or searching through a database of images [Mil+17].

A popular approach to these tasks has been to observe humans engaging in dialogs like the ones we would like to automate and then train agents to mimic these human dialogs [Das+17a; Lew+17]. Mimicking human dialogs allows agents to generate interpretable language (*i.e.*, meaningful English, not gibberish). However, these models are typically fragile and generalize poorly to new tasks. As such, each new task requires collecting new human dialogs, which is a



**Fig. 4.1.:** (Top - 2 pools) We train our questioner to ask questions that can discriminate between pairs of images by mimicing questions from the VQAv2 dataset. (Bottom - 1 pool) Our proposed model generalizes to new settings in a way that humans can understand without additional language supervision (*i.e.*, without dialog).

laborious and costly process often requiring many iterations before high quality dialogs are elicited.

A promising pragmatic alternative is to use goal completion as a supervisory signal to adapt agents to new tasks. That is, after training dialog agents to mimic human dialogs for one task, fine-tune them on a new task by simply rewarding the agents for solving the task regardless of the dialog’s content. This approach can indeed improve task performance, but language quality suffers even for similar tasks. It tends to drifts from human language, becoming ungrammatical and loosing human interpretable semantics – sometimes even turning into unintelligible code. Though bots might understand it, humans cannot, so humans will not be able to use it either. Both effects have been observed in prior dialog work [Das+17a; Lew+17].

In this work, we consider an image guessing game as a test-bed for balancing task performance and language drift. Our Dialog without Dialog (DwD) task requires agents to generalize from single round visual question generation with full supervision to a multi-round dialog based image guessing game without direct language supervision. Specifically, as illustrated in (Fig. 4.1) (top), agents

are trained to mimic human-generated, visually-grounded questions that when answered can discern which of two images is secretly indicated to the answerer. We then develop techniques to transfer these agents to a multi-round, QA-based image guessing game over pools of various sizes, difficulties, and even image domains.

To solve this task we propose an architecture for the questioner agent, Q-bot, that decomposes generating question intent from the words used to express that intent. It does this by introducing a discrete latent representation that is the only input to the language decoder. We pair this with an incremental learning curriculum that adapts the single round Q-bot to dialog in stages – first learning simply to follow the dialog and then to influence question intention.

We show that our model can be fine-tuned to increase task performance while maintaining human interpretable language. To measure interpretability we take a two pronged approach, getting humans to evaluate the fluency and relevance of questions generated by our model on one hand and using automatic measures of fluency, relevance, and diversity to help scale our analysis. To summarize, our contributions are:

- We propose the Dialog without Dialog (DwD) task, where the goal is to balance task performance with human interpretability in a multi-round image guessing game while only using non-dialog language supervision and task level dialog feedback.
- We propose a questioner model for DwD that factorizes task-specific and task-agnostic components using discrete latent variables and an incremental training regime.
- We perform extensive experiments that consider tasks increasingly distant from the one in which we have language supervision. Our baselines general-

ize poorly to these tasks, loosing interpretability or task performance, but our model achieves a better balance of task performance and interpretability.

## 4.2 Dialog-based Image Guessing Game

Our objective is to examine how to transfer grounded language models from one task to another by training agents only to maximize task success. We consider an image-guessing communication game as the context for our experiments. In this section, we introduce this game and a model for this multi-round dialog task. In the following sections, we will discuss how to train such a model using non-dialog data.

### 4.2.1 Game Definition

We consider a conceptually simple image guessing game demonstrated in (Fig. 4.1). In each episode, one agent (A-bot in red) secretly selects an image  $y$  (starred) from an image pool (in the dashed green box). The other agent (Q-bot in green) must identify this image by executing a multi-round question-answer based dialog with A-bot. To succeed, Q-bot will need to understand the image pool, generate discriminative questions, and interpret the answers A-bot provides to identify A-bot’s selected image.

At a high-level functional view, we can consider the dialog as following a simple structure. At each round  $r$ , Q-bot observes the pool  $\mathcal{I} = \{I_1, \dots, I_P\}$  and dialog history  $q_0, a_0, \dots, q_{r-1}, a_{r-1}$  and produces a question

$$q_r = \text{QBot.Ask}(\mathcal{I}, q_0, a_0, \dots, q_{r-1}, a_{r-1}). \quad (4.1)$$

Given this question  $q_r$ , A-bot provides an answer  $a_r$  based on its selected image  $I_y$ :

$$a_r = \text{ABot.Answer}(I_y, q_r) \quad (4.2)$$

Once Q-bot receives the answer from A-bot, it makes a prediction  $\hat{y}_{r+1}$  about the target image:

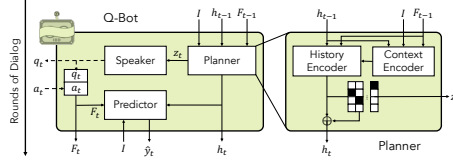
$$\hat{y}_r = \text{QBot.Predict}(\mathcal{I}, q_0, a_0, \dots, q_r, a_r) \quad (4.3)$$

where the task performance of Q-bot can be calculated by comparing  $\hat{y}_r$  and  $y$ .

**Comparison to GuessWhich..** Das et al. [Das+17a] presented a similar dialog-based guessing game called GuessWhich. In GuessWhich, Q-bot initially observes a caption describing A-bot’s selected image and must predict the selected image’s features to retrieve it from a large, fixed pool of images. The inclusion of the caption leaves little room for the dialog to add information [Mir+17] and the fixed-pool would not enable us to inspect how Q-bot’s behavior generalizes to different pools. As described above, we drop both these assumptions to enable our analysis.

## 4.2.2 Modelling A-bot

In this work, we focus primarily on Q-bot agent rather than A-bot. We set A-bot to be a standard visual question answering agent, specifically the Bottom-up Top-down [Ten+17] model; however, we do make one modification. Q-bot may generate questions that are not well grounded in A-bot’s selected image (though they may be grounded in other pool images) – e.g. asking about a surfer when none exists. To enable A-bot to respond appropriately, we augment A-bot’s answer space with a `Not Relevant` token. We augment every image with an



**Fig. 4.2.:** A single round of our Q-bot which decomposes into the modules described in Section 4.2.3. This factorization allows us to fine-tune just the intention of the model for task performance, limiting language drift.

additional, randomly-sampled question and set `Not Relevant` as its target answer. A-bot is trained independently from Q-bot on the VQAv2 dataset and then frozen.

### 4.2.3 Modelling Q-bot

We conceptualize Q-bot as having three major tasks: encoding the state of the game to decide what to ask about, actually formulating this intent in language, and making predictions about A-bot’s selection. Respectively, these correspond to planner, speaker, and predictor modules. As we focus on language transfer across tasks, we make fairly standard design choices here.

**Pool & Image Encoding.** We represent the  $p$ th image  $I_p$  of the pool as a set of  $B$  bounding boxes such that  $I_p^b$  is the embedding of the  $b$ -th box following [And+18]. Note that we do not assume prior knowledge about the size or composition of the pool.

#### Planner

The planner’s role is to encode the dialog context (image pool and dialog history) and decide what to ask about in each round. To limit clutter, we denote the QA pair at round  $r$  as a ‘fact’  $F_r = [q_r, a_r]$ .



**Context Encoder.** Given the prior dialog state  $h_{r-1}$ ,  $F_{r-1}$ , and image pool  $\mathcal{I}$ , the context encoder performs hierarchical attention to identify image regions in the pool that are most relevant for generating the next question. As we describe in appendix A,  $F_{r-1}$  and  $h_{r-1}$  to query the image to compute an attention distribution over both set of images ( $\alpha_j$ ) and  $P$  distributions over the bounding boxes in each image ( $\beta_j^i$ ). The overall image encoding  $\hat{v}_r$  at round  $r$  is computed as

$$\hat{v}_r = \sum_{j=1}^P \sum_{i=1}^B \alpha_j \beta_j^i v_j^i \quad (4.4)$$

where both image and region attentions are combined. We leave the details on computing these attention distributions to the appendix to conserve space. We note that this mechanism is agnostic to the pool size.

**History Encoder.** To track the state of the game, the planner applies an LSTM-based history encoder that takes  $\hat{v}_r$  and  $F_r$  as input and produces an intermediate hidden state  $h_{r+1}$ . Here  $h_{r+1}$  includes a compact representation of question intent and dialog history, providing a differentiable connection between the intent and final predictions through the dialog state.

**Question Policy.** The question policy transforms  $h_{r+1}$  to a question representation  $z_r$  that will be passed to the speaker model to generate the actual question text. In some sense,  $z_r$  corresponds to the “intent” of the question (e.g. checking the existence of surfers) that triggers the speaker to produce corresponding text (e.g. “Is anyone surfing?”). A default choice for  $z_r$  is identity function (i.e.,  $z_r = h_{r+1}$ ). Later we explore choices where  $z_r$  is a random variable (continuous or discrete) parameterized by  $h_{r+1}$ .

## Speaker

Given an intent  $z_r$ , the speaker generates a natural language question. We model the speaker as a standard LSTM-based decoder with an initial hidden state equal to  $z_r$  (or an embedding of  $z_r$  for discrete  $z_r$ ).

## Predictor

The predictor uses the planner’s hidden state to guess which image A-bot has selected. The predictor takes a concatenation  $F = [F_1, \dots, F_{r+1}]$  of fact embeddings and the dialog state  $h_{r+1}$  and computes an attention pooled feature  $\hat{F}$  using  $h_{r+1}$  as attention context. A score is then computed for each image in the pool based on the image features, the pooled representation, and the dialog state (see appendix for full model details). These scores are normalized via a softmax to predict the target image. The model can then be trained end-to-end to minimize a cross-entropy loss on this prediction. Note the model is agnostic to the pool size.

## 4.3 Dialog without Dialog

Aside from some abstracted details, the game setting and model presented in the previous section could be trained without any further information – a pool of images could be generated, A-bot could be assigned an image, the game could be rolled out for arbitrarily many rounds, and Q-bot could be trained to predict the correct image given A-bot’s answers. While conceptually possible, there is an obvious shortcoming – it would be nigh impossible for Q-bot to learn to

produce interpretable questions. Nobody discovers French. They have to learn it.

At the other extreme – representing standard practice in dialog problems – humans could be recruited to perform this image guessing game and provide dense supervision for what questions Q-bot should ask to perform well at this specific task. However, this suggests a machine learning paradigm that requires collecting language data for every new task. Aside from being costly, it is intellectually dissatisfying for agents’ knowledge of natural language to be so inseparably intertwined with individual tasks. After all, one of the greatest powers of language is the ability to use it to communicate about many different problems.

In this section, we consider a middle-ground – training our agents with single-shot question answering data and then learning an agent that can carry on our task-driven dialog without further supervision.

#### 4.3.1 Stage 1: Language Pre-training

We want Q-bot’s language to be interpretable – in this paper we take that to mean it should be understandable by and semantically meaningful to humans, so it has to be something like a meaningful subset of a known human language. To pre-train the model to use interpretable human language, we design a supervised learning task for a single-round version of our game.

We leverage the VQAv2 [Goy+17] dataset as our language source to learn how to ask interpretable questions. By construction, for each question in VQAv2 there exists at least one image pair which are visually similar but have different ground truth answers to the question. This somewhat mirrors our dialog game

– the image pair is the pool, the question is guaranteed to be discriminative, and we can provide an answer depending on A-bot’s selected image. We can view this as a special case of our game that is fully supervised but contains only a single round of dialog. We can then train our Q-bot to mimic the human question (e.g. via cross-entropy teacher forcing) and to predict the correct image given the ground-truth answer.

### 4.3.2 Stage 2: Transferring to Dialog

The VQA dataset contains simple questions about images, but they are not aimed at accomplishing our image guessing task. Consequently, the goal of Dialog without Dialog is to transfer this learned language understanding to new tasks and demonstrate generalization in terms of interpretability *and* task performance across many task variations (e.g. multiple rounds of conversation and new pools of images).

As an initial setting, we could take the pre-trained weights from Stage 1 and simply fine-tune for our full image guessing task. However, this agent would face a number of challenges. It has never had to model multiple steps of a dialog. Further, while following the task objective of predicting A-bot’s selected image, there is little to encourage Q-bot to continue producing interpretable language. We consider a number of modifications to address these problems.

**Discrete Intention  $z$  Representation..** Rather than a continuous vector passing from the question policy to the speaker, we consider a discrete random variable. Specifically, we consider a representation composed of  $N$   $K$ -way Concrete variables [MMT17] so  $z_n \in [0, 1]^K$  is a distribution over  $K$  objects.

We learn a linear transformation from the intermediate dialog state  $\bar{h}_r$  to a set of logits  $l_{Kn:K(n+1)-1}^z$  for each variable  $n$  in  $z$ :

$$l_{Kn:K(n+1)-1}^z = \text{LogSoftmax}(h_{Kn:K(n+1)-1}) \forall n \quad (4.5)$$

This parameterizes encoder distribution  $p(z_r)$ .

To provide input to the speaker,  $z_r$  is embedded using a learned dictionary of embeddings. In our case each variable in  $z$  has a dictionary of  $K$  learned embeddings. The value of  $z_n$  ( $\in \{1, \dots, K\}$ ) picks one of the embeddings for each variable and the final representation simply sums over all variables:

$$e_z = \sum_{n=0}^{N-1} E_n^z(z_n). \quad (4.6)$$

**VAE Pre-training.** When using this representation for the intent, we train Stage 1 by replacing the likelihood with an ELBO loss to restrict information flow through  $z$ . This requires an encoder and a decoder. The decoder is the speaker and the encoder is a new module  $q(z|q_0, \mathcal{I})$  that forms a conditional distribution over  $z$ . For the encoder we use a version of the previously described context encoder that uses just the question  $q_0$  as attention query and parameterizes this Concrete distribution with a linear transformation of the resulting hidden state. The resulting ELBO loss is like the Full ELBO described (but not implemented) in [ZXE19]:

$$\mathcal{L} = E_{z \sim q(z|q_0, \mathcal{I})} [\log p(\text{speaker}(z))] \quad (4.7)$$

$$+ \frac{1}{N} \sum_{n=0}^{N-1} D_{KL} [q(z_n|q_0, \mathcal{I}) || \mathcal{U}(K)] \quad (4.8)$$

The first term encourages the encoder to mimic the VQA question. The second term pushes the distribution of  $z$  close to a  $K$ -way uniform prior, which forces  $z$  to only carry relevant information. Combined, the first two terms form an ELBO on the question likelihood given the image pool [JGP17; Kai+18; ZXE19].

**Fixed Speaker.** Since the speaker contains only lower level information about how to generate language, we freeze it during task transfer. We want only the high level ideas represented by  $z$  and the predictor which receives direct feedback to adapt to the new task. If we updated the speaker then it could overfit its language to the sparse feedback available in each new setting.

**Adaptation Curriculum.** As the pre-trained model has never had to keep track of dialog contexts beyond the first round, we fine-tune in two stages. In **Stage 2.A** we fix the Context Encoder and Question Policy parts of the Planner so the model can learn to track dialog effectively without trying to generate better dialog at the same time. This stage takes 20 epochs to train. Once Q-bot learns how to track dialog we update the entire planner in **Stage 2.B** for 5 epochs.<sup>1</sup>

## 4.4 Experiments

### 4.4.1 Settings

We consider experimental settings which test generalization along four dimensions: dialog round, pool type, pool size, and image domain. We can control the difficulties of the proposed DwD task by setting the number of dialog round, number of type of images in the pool and whether the task is operate on a different image domain. We consider three image sources – COCO [Lin+14b],

---

<sup>1</sup>We find that 5 epochs stops training early enough to avoid the significant overfitting that can otherwise occur.

CUB [Wah+11], and AWA [Xia+18]. We vary pool size to be either 2 or 9 images either randomly selected or a contrasting pair (the synthetic VQA pools from Stage 1, only defined for VQA pool size 2). Unless specified, performance is reported for Q-bot’s final guess at the last round.

#### 4.4.2 Metrics

We consider metrics addressing both **Task** performance and **Language** quality. While task performance is straightforward, language quality is harder to measure. We use multiple metrics including human evaluations reported in Section 4.4.4.

**Task - Guessing Game Accuracy via A-bot..** The point of transfer is to improve task performance so we report the accuracy of Q-bot’s guess at the final round of dialog.

**Language - Question Relevance via A-bot..** To be human understandable, the generated questions should be relevant to at least one image in the pool. We measure question relevance as the maximum question-image relevance across the pool as measured by A-bot, i.e.  $1 - p(\text{Not Relevant})$ . We note that this is only a proxy for actual question relevance as A-bot may report `Not Relevant` erroneously if it fails to understand Q-bot’s question; however, in practice we find A-bot does a fair job in determining relevance. We also provide human relevance judgements in Section 4.4.4.

**Language - Fluency via Perplexity.** To evaluate Q-bot’s fluency, we train an LSTM-based language model on the entire corpus of questions in VQA. This allows us to evaluate the perplexity of the questions generated by Q-bot for dialogs on its new tasks. Lower perplexity indicates the generated questions are

similar to VQA questions in terms of syntax and content. Questions generated for the new tasks could have lower perplexity because they have drifted from English or because different things must be asked for the new task, so lower perplexity is not always better [TOB15].

**Language - Diversity via Distinct  $n$ -grams.** This considers the set of all questions generated by Q-bot across all rounds of dialog on the val set. It counts the number of  $n$ -grams in this set,  $N_n$ , and the number of distinct  $n$ -grams in this set,  $D_n$ , then reports  $\frac{N_n}{D_n}$  for each value of  $n \in \{1, 2, 3, 4\}$ . Note that instead of normalizing by the number of words as in previous work [Ash+18; Li+15], we normalize by the number of  $n$ -grams so that the metric represents a percentage for values of  $n$  other than  $n = 1$ . Generative language models frequently produce safe standard outputs [Ash+18], so diversity is a sign this problem is decreasing, but diversity by itself does not make language meaningful or useful.

### 4.4.3 Results

**Baselines..** We compare our proposed approach to two baselines – **Stage 1** and **Non-Var Cont** – each ablating some aspects of our design choices. The **Stage 1** baseline is our model after the single-round fully-supervised pretraining. Improvements over this model represent gains made from task-based fine-tuning. The **Non-Var Cont** baseline is our model under standard encoder-decoder dialog model design choices – i.e. a continuous latent variable, maximum-likelihood pre-training, and fine-tuning the speaker model.

**Results.** (Tab. 4.1) presents results for our model and baselines in different settings. Starting from the first setting and moving downward, agents are tasked with generalizing further and further from their source data – from setting A



**Tab. 4.1.:** Performance of our models and baselines in different experimental settings. From setting A to setting F, agents are tasked with generalizing further from the source data. Our method strikes a balance between guessing game performance and interpretability.

			Accuracy $\uparrow$	Perplexity $\downarrow$	A-bot Relevance $\uparrow$	Diversity $\uparrow$
<b>VQA</b> 2 Contrast 1 Round	A1	Stage 1	0.73	2.62	0.87	0.50
	A2	Non-Var Cont	0.71	10.62	0.66	<b>5.55</b>
	A3	Ours	<b>0.82</b>	<b>2.6</b>	<b>0.88</b>	0.54
<b>VQA</b> 2 Contrast 5 Rounds	B1	Stage 1	0.67	2.62	0.87	0.50
	B2	Non-Var Cont	0.74	10.62	0.66	<b>5.55</b>
	B3	Ours	<b>0.87</b>	<b>2.60</b>	<b>0.88</b>	0.54
<b>VQA</b> 2 Random 5 Rounds	C1	Stage 1	0.64	<b>2.64</b>	0.75	1.73
	C2	Non-Var Cont	0.86	16.95	0.62	<b>8.13</b>
	C3	Ours	<b>0.95</b>	2.69	<b>0.77</b>	2.34
<b>VQA</b> 9 Random 9 Rounds	D1	Stage 1	0.18	2.72	<b>0.77</b>	1.11
	D2	Non-Var Cont	<b>0.78</b>	40.66	<b>0.77</b>	<b>2.57</b>
	D3	Ours	0.53	<b>2.55</b>	0.75	0.95
<b>AWA</b> 9 Random 9 Rounds	E1	Stage 1	0.47	2.49	<b>0.96</b>	0.24
	E2	Non-Var Cont	0.48	12.56	0.64	<b>2.21</b>
	E3	Ours	<b>0.74</b>	<b>2.41</b>	<b>0.96</b>	0.28
<b>CUB</b> 9 Random 9 Rounds	F1	Stage 1	0.36	2.56	<b>1.00</b>	0.04
	F2	Non-Var Cont	0.38	20.92	0.47	<b>2.16</b>
	F3	Ours	<b>0.74</b>	<b>2.47</b>	<b>1.00</b>	0.04

which mimics the human data pretraining to setting F where agents must carry on a nine round dialog about 9 images containing only different bird species. Our final model uniformly performs well on both task performance and language fluency across different settings in terms of the automatic evaluation metrics (see bolded results). Other key findings are:

**Ours vs. Stage 1:** To understand the relative importance of the proposed stage 2 training which transferring to dialog for DwD task, we compared the task accuracy of our model with that of Stage 1. In setting A which matches the training regime, our model outperforms Stage 1 by 9% on task performance. As the tasks differ in settings B-F, we see further gains with our model consistently outperforming Stage 1 by 20-38%. Despite these gains, our model maintains similar language perplexity, A-bot relevance, and diversity.

**Ours vs. Non-Var Cont:** Our discrete latent variable, variational pre-training objective, and fixed speaker also play a important roles in avoiding language drift. Compared to the Non-Var Cont model without these techniques, our model achieves over 4x lower perplexity and 10-53% better A-bot Relevance. Our model also improves the averaged accuracy over the Non-Var Cont model, which means more interpretable language also improves the task performance. Note that Non-Var Cont has 2-100x higher diversity compared to our model, since the language is shifted away from English (and towards gibberish).

#### **Game Variations:**

- **Dialog Rounds:** Longer dialogs (more rounds) achieve better accuracy (A3 vs B3).
- **Pool Type:** Random pools are easier compared to contrast pool (B3 vs C3 accuracy), however, language fluency and relevance drop on the random pools (B3 vs C3 perplexity and a-bot relevance).

**Tab. 4.2.:** Human evaluation of language quality – question fluency (top) and relevance (bottom). Each row compares a pair of agent-generated questions, asking users which (or possibly neither) is more fluent/relevant. The values report the percentage of times the option represented by that column was chosen.

	Neither	Stage 1	Non-Var Cont	Ours
Stage 1 vs Non-Var Cont	31.7%	48.1%	20.2%	–
Stage 1 vs Ours	49.0%	26.2%	–	24.8%
Non-Var Cont vs Ours	32.7%	–	17.9%	49.4%
Stage 1 vs Non-Var Cont	19.6%	48.8%	31.7%	–
Stage 1 vs Ours	25.0%	38.4%	–	36.6%
Non-Var Cont vs Ours	22.0%	–	30.2%	47.8%

- **Image Source:** CUB and AWA pools are harder compared to COCO image domain (D3 vs E3 vs F3). Surprisingly, our models maintains similar perplexity and high a-bot relevance even on these out-of-domain image pools. The Stage 1 and Non-Var Cont baselines generalize poorly to these different image domains – reporting task accuracies nearly half our model performance.

#### 4.4.4 Human Studies


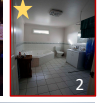










In addition to the automatic metrics, we also evaluate our models through human studies. Specifically, we use workers (turkers) on Amazon Mechanical Turk to evaluate the relevance, fluency, and task performance of our models. We discuss each study below.

**Human Study for Question Relevance..** To get a more accurate measure of question relevance, we asked humans to evaluate questions generated by our model and the baselines (Stage 1 & Non-Var Cont). We curated 300 random, size 4 pools where all three models predicted the target correctly at round 5. For a random round, we show turker’s the questions from a pair of models and ask "Which question is most relevant to the images?" Answering the question is a

forced choice between three options: either of the pair of models or an “equally relevant” option. More details including an example of the interface can be found in appendix C. (Tab. 4.2) (bottom) shows the frequency with which each option was chosen for each model pair. Our model is considered more relevant than the Non-Var Cont model (47.8% vs. 30.2% preference) and about the same as the Stage 1 model (36.6% vs. 38.4% preference).

**Human Study for Fluency..** We also evaluate fluency by asking humans to compare questions. In particular, we presented the same pairs of questions to turkers as in the relevance study, but this time we did *not* present the pool of images and asked them "Which question is more understandable?" As before, there was a forced choice between two models and an “equally understandable” option. This captures fluency because humans are more likely to understand grammatically correct. (Tab. 4.2) (top) shows the frequency with which each option was chosen for each model pair. Our model is considered more fluent than the Non-Var Cont model (49.4% vs. 17.9% preference) and about the same as the Stage 1 model (49.0% neither question more fluent).

**Human Study for Task Performance..** What we really want is for humans to be able to collaborate with bots to solve tasks. Therefore, the most direct evaluation of our the DwD task is to have humans interact dynamically with Q-bot. We implemented an interface that allowed turkers to interact with Q-bot in real time. Q-bot asks a question. A human answers it. Q-bot asks a new question in response to the human answer and the human responds to that question. After the 4th answer Q-bot makes a guess about which target image the human was answering based on. Our interface is described in section C of the supplement. We perform this study for the same pools for each model and find our approach achieves an accuracy of 69.39% – significantly higher than Non-Var Cont at 44.90% and Stage 1 at 22.92%. This study shows that our model learns a strategy for this task that is amenable to human-AI collaboration.

	Non-Var Cont	Stage 1	Ours
 1  2  3  4	<b>Q0:</b> what is the boy in? not relevant : A0 P0: 4 <b>Q1:</b> how many objects can be breadsticks? 2 : A1 P1: 1 <b>Q2:</b> sweetest meters what is the color? white : A2 P2: 4 <b>Q3:</b> diving what day is the cabinet? oval : A3 P3: 2 <b>Q4:</b> equestrian pads what can be seen ?	<b>Q0:</b> is there a reflection? no : A0 P0: 2 <b>Q1:</b> what fruit is walking across the right? not relevant : A1 P1: 2 <b>Q2:</b> what is bright in the corner? light : A2 P2: 2 <b>Q3:</b> is it time? not relevant : A3 P3: 3 <b>Q4:</b> is there a cat in this photo?	<b>Q0:</b> What color are the wheels ? not relevant : A0 P0: 4 <b>Q1:</b> what is the color of the white fence ? not relevant : A1 P1: 1 <b>Q2:</b> how many people in the room? white : A2 P2: 4 <b>Q3:</b> which room is this ? bathroom : A3 <b>Q4:</b> is this picture taken during a day?
 1  2  3  4	<b>Q0:</b> what color is the photo? not relevant : A0 P0: 4 <b>Q1:</b> what is the on the bottom person? not relevant : A1 P1: 4 <b>Q2:</b> what shape is this light? not relevant : A2 P2: 4 <b>Q3:</b> what shape is the train? not relevant : A3 P3: 4 <b>Q4:</b> what shape of this?	<b>Q0:</b> how many legs are visible? 2 : A0 P0: 2 <b>Q1:</b> how many different pillows are in the pic? not relevant : A1 P1: 3 <b>Q2:</b> what is the animal that is next to the blue animal's leg? bear : A2 P2: 4 <b>Q3:</b> what number is on the boogie head? not relevant : A3 P3: 3 <b>Q4:</b> is this animal hungry?	<b>Q0:</b> what kind of animal is this? Polar bear : A0 P0: 4 <b>Q1:</b> how many little dogs are laying around? 0 : A1 P1: 4 <b>Q2:</b> what color is the bear? white : A2 P2: 4 <b>Q3:</b> what is the animal holding? nothing : A3 P3: 4 <b>Q4:</b> can the animal be seen in the water?
 1  2  3  4	<b>Q0:</b> what color is the photo? gray : A0 P0: 3 <b>Q1:</b> is the boy's collar on the right? not relevant : A1 P1: 3 <b>Q2:</b> what color is the thing? black : A2 P2: 3 <b>Q3:</b> what is the color? black : A3 P3: 3 <b>Q4:</b> what is the first?	<b>Q0:</b> what is on the bowl? bird : A0 P0: 1 <b>Q1:</b> how is the sitting on water? sand : A1 P1: 4 <b>Q2:</b> what kind of birds are these? crow : A2 P2: 4 <b>Q3:</b> what is the bird eating? nothing : A3 P3: 3 <b>Q4:</b> does the bird have a sheep 's tail toy?	<b>Q0:</b> what is behind the bird ? sand : A0 P0: 4 <b>Q1:</b> what is the color of the collar? not relevant : A1 P1: 4 <b>Q2:</b> what kind of bird is in the image ? crow : A2 P2: 3 <b>Q3:</b> what kind of bird is this ? crow : A3 P3: 3 <b>Q4:</b> what is the bird sitting on ?

**Fig. 4.3.:** Qualitative comparison of dialogs generated by our model with those generated by Non-Var Cont and Stage 1 baselines. Top / middle /bottom rows are image pool from COCO / AWA / CUB images respectively. Our model pretrained on VQA (COCO image) generates more interpretable questions for the DwD task which is semantic meaning and generalize well to out-of-domain images.

This is in contrast to prior work [Cha+17] that showed that improvements captured by task-trained models for similar image-retrieval tasks did not transfer when paired with human partners.

#### 4.4.5 Qualitative Results

Figure 4.3 shows example outputs of Non-Var Cont baseline, Stage 1 model and our proposed models on three different image sources – COCO, AWA and CUB datasets. We can see that COCO images contains varieties of concepts while AWA images contains on different animals and CUB images contains on different species of birds. The A-bot is not accurate, which introduces noisy signals for Q-bot to learn the DWD tasks. Compared with the baselines, our approach asks more relevant and interpretable questions in the dialog.

#### 4.4.6 Model Ablations

We investigate the impact of our modelling choices from Section 4.3. In (Tab. 4.3) we report the mean of all four automated metrics averaged over pool sizes, pool sampling strategies, and datasets.<sup>2</sup>Next we explain how we vary each of these model dimensions

- Our 128 4-way Concrete variables require 512 logits (**Discrete**). Thus we compare to the standard Gaussian random variable common throughout VAEs with 512 dimensions (**Continuous**). This just removes the KL term ((4.8)).
- In both discrete and continuous cases we train with an ELBO loss (**ELBO**), so we compare to a maximum likelihood only model (**MLE**) that uses an identity function as in the default option for the Question Policy (see Section 4.2.3).
- We consider checkpoints after each step of our training curriculum: **Stage 1**, **Stage 2.A**, and **Stage 2.B**. For some approaches we skip Stage 2.A and go straight to fine-tuning everything except the speaker as in Stage 2.B. This is denoted by **Stage 2**.
- We consider 3 variations on how the speaker is fine-tuned. The first is our proposed approach of fixing the speaker (**Fixed**). The next fine-tunes the speaker (**Fine-tuned**). To evaluate the impact of fine-tuning we also consider a version of the speaker which can not learn to ask better questions by using a parallel version of the same model (**Parallel**). This last version will be described more below.

**Discrete Outperforms Continuous** *z.*. By comparing our model in row 1 of (Tab. 4.3) to row 7 we see that our discrete model outperforms the corresponding

---

<sup>2</sup>This includes 10 settings: {random 2, 4, 9 pools }  $\times$  {VQA, AWA, CUB} and 2 contrasts pools on VQA

**Tab. 4.3.:** Various ablations of our training curriculum.

	$z$ Structure	Loss	Curriculum	Speaker	Accuracy	Perplexity	Relevance	Diversity
1	Discrete	ELBO	Stage 2.B	Fixed (Ours)	0.81	2.57	0.89	0.86
2	Discrete	ELBO	Stage 2	Fine-tuned	0.82	2.54	0.85	0.59
3	Discrete	ELBO	Stage 2	Parallel	0.78	2.60	0.88	0.73
4	Discrete	ELBO	Stage 1	Fixed	0.72	2.60	0.91	0.48
5	Discrete	ELBO	Stage 2.A	Fixed	0.80	2.59	0.89	0.81
6	Discrete	ELBO	Stage 2	Fixed	0.80	2.53	0.85	0.62
7	Continuous	ELBO	Stage 2.B	Fixed	0.75	2.45	0.66	0.23
8	Continuous	MLE	Stage 2.B	Fixed	0.78	4.27	0.83	4.33

continuous model in terms of task performance (higher Accuracy) and about matches it in interpretability (similar Perplexity and higher Relevance). This may be a result of discreteness constraining the optimization problem to prevent overfitting and is consistent with previous work that used a discrete latent variable to model dialog [ZXE19].

**Stage 2.B Less Important than Stage 2.A.** Comparing rows 4, 5, and 1 of (Tab. 4.3), we can see that each additional step, Stage 2.A (row 4  $\rightarrow$  5) and Stage 2.B (row 5  $\rightarrow$  1), increases task performance and stays about the same in terms of interpretability. However, most gains in task performance happen between Stage 1 and Stage 2. This indicates that improvements in task performance are mainly from learning to incorporate information over multiple rounds of dialog.

**Better Predictions, Slightly Better Questions.** To further investigate whether Q-bot is asking better questions or just understanding dialog context for prediction better we considered the **Parallel** speaker model. This model loaded two copies of Q-bot, A and B both starting at Stage 1. Copy A was fine-tuned for task performance, but every  $z$  it generated was ignored and replaced with the  $z$  generated by copy B, which was not updated at all. The result was that copy A of the model could not incorporate dialog context into its questions any better than the Stage 1 model, so all it could do was track the dialog better for prediction purposes. By comparing the performance of copy A (row 3 of

(Tab. 4.3)) to our model (row 1) we can see a 3 point different in accuracy, so the question content of our model has improved after fine-tuning, but not by a lot. Most improvements are from dialog tracking for prediction (row 3 accuracy is much higher than row 4 accuracy).

**Fine-tuned Speaker.** During both Stage 2.A and Stage 2.B we fix the Speaker module because it is intended to capture low level language details and we do not want it to change its understanding of English. Row 2 of (Tab. 4.3) does not fix the Speaker during Stage 2 fine-tuning. Instead, it uses each softmax at each step of the LSTM decoder to parameterize one Concrete variable [JGP17] per word. This allows gradients to flow through the decoder during fine-tuning, allowing the model to tune low-level signals. This is similar to previous approaches which either used this technique [Lu+17] or REINFORCE [Das+17a] This model is competitive with DWD in terms of task performance. However, when we inspect its output we see somewhat less interpretable language.

**Variational Prior Helps Interpretability.** We found the most important factor for maintaining interpretability to be the ELBO loss we applied during pre-training. Comparing the continuous Gaussian variable (row 7) to a similar hidden state (row 8) trained without the prior term (4.8) we see drastically different perplexity and diversity. Perplexity and diversity drop because the model has drifted far from English. This is similar to the effect in the Non-Var Cont, which is the model from row 8 with a fine-tuned speaker.

## 4.5 Related Work

We use a visual reference game to study question generation, and in particular we are interested in interpretable and efficiently learning language. This interest is mainly inspired by problems encountered when using models comparable to



the Stage 1 baseline from Section 4.4.3. In [Lew+17] a dataset is collected with question supervision then fine-tuning is used in an attempt to increase task performance, but the resulting utterances are uninterpretable. Similarly, [Das+17a] takes a very careful approach to fine-tuning for task performance but finds that language also diverges, becoming difficult for humans to understand.

**Visual Question Generation..** Other approaches like [Mis+18] and [Yan+18] also aim to ask questions with limited question supervision. They give Q-bot access to an oracle to which it can ask any question and get a good answer back. This feedback allows these models to ask questions that are more useful for teaching A-bot [Mis+18] or generating scene graphs [Yan+18], but they require a domain specific oracle and do not take any measures to encourage interpretability. We are also interested in generalizing with limited supervision, using a standard VQAv2 [Goy+17] trained A-bot as a flawed oracle, but we focus on maintaining interpretability of generated questions and not just their usefulness.

**Latent Action Spaces..** Of particular interest to us is a line of work that uses represents dialogs using latent action spaces [ZLE18; ZE18; YL17; Wen+17; Ser+16; YL17; Hu+19; Kan+19; Ser+17; WAZ17]. Recent work uses these representations to discover interpretable language [ZLE18] and to perform zero-shot dialog generation [ZE18], though neither works consider visually grounded language as in our approach. Most relevant is [ZXE19], which focuses on the difference between word level feedback and latent action level feedback. Like us, they use a variationally constrained latent action space (like our  $z$ ) to generate dialogs and find that by providing feedback to the latent actions instead of the generated words (as opposed to the approaches in [Das+17a] and [Lew+17]) they achieve better dialog performance. Our variational prior is similar to the Full ELBO considered in [ZXE19], but we consider generalization from non-dialog data and generalization to new modalities.

**Reference Games..** The task we use to study question generation follows a body of work that uses reference games to study language and its interaction with other modalities [Lew69]. Our particular task is most similar to those in [Vri+16] and [Cha+17]. In particular, [Vri+16] collects a dataset for goal oriented visual dialog using a similar image reference game and [Cha+17] uses a similar guessing game we use to evaluate how well humans can interact with A-bot.

## 4.6 Conclusion

In this paper we proposed the Dialog without Dialog (DwD) task along with a model designed to solve this task and an evaluation scheme that takes its goals into account. The task is to build a dialog agent that generates meaningful and useful dialogs without dialog level language supervision. This balance is hard to strike, but our proposed model manages to strike it. We find it helps to represent dialogs with a discrete latent variable and carefully transfer language information via multi-stage training. While baseline models either perform well at new tasks through fine-tuning or maintain interpretability, our model achieves the goal of DwD by doing both. We hope both our task and our model help inspire useful dialog agents that can also interact with humans.

## Conclusion

” *You can build crystal palaces of thought,  
working from first principles, then climb up  
inside them and pull the ladder up behind  
you.*

— Maciej Cegłowski [Ceg16]

Neural networks have supported significant progress on perceptual AI tasks, but there are still challenges scaling them to increasingly complex tasks. As task complexity increases so does the cost of collecting data and the need for data to feed these hungry models. We showed how the problem can be alleviated by adding new inductive biases centered around the idea of disentanglement to our models. The thesis investigated three ways of disentangling neural net representations and showed that in each case better generalization resulted.

In Chapter 2 we considered statistical independence, or redundancy, as a notion of disentanglement. We learned more general image classifiers by adding a loss that discouraged redundant image representations.

In Chapter 3 we considered compositionality as a notion of disentanglement and we were interested in whether neural nets could discover compositional language. We found that the language discovered by neural nets tended to be non-compositional, but that we could improve its compositionality by adding a

context where language could be transferred across generations in a population of agents. This resulted in disentangled language that could better generalize to new compositionally novel examples.

Finally, in Chapter 4 we disentangled intention from language in question asking agents. Here the goal was to achieve high accuracy on an image guessing task while maintaining high language quality. We proposed a model that disentangles intention from language and showed that baselines can either solve the image guessing game or maintain language quality, but only our model does both.

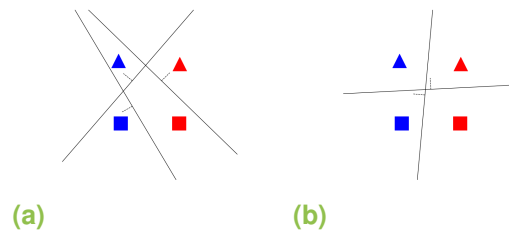
As a high level intuition, disentanglement is not directly useful. It needs to be implemented in a design mechanism like those showcased above, which is a non-trivial step. However, by highlighting disentanglement as a perspective around which to focus neural net designs we hope this thesis can inspire new implementations of the high level intuition and can help practitioners connect different research directions in new ways.

## 5.1 Implications and Future Work

### 5.1.1 Inductive Biases

Together, these chapters illustrate different instantiations of an inductive biases. Yet it is unclear what role inductive biases like these have to play, and in particular how important they are for designing neural networks. Often increasing the amount of training data results in a similar improvement in learning performance. But despite the simplicity of that approach, it can have substantial disadvantages:

1. Data collection is inefficient, especially compared to the ease with which we can add many inductive biases to our model. For example, `DeCov` may only require changing a couple lines of code in modern frameworks.
2. Furthermore, we do not generally know how much data is needed to get the desired level of generalization. Sometimes an order of magnitude more data may offer significant improvements and other times gains from the same approach can be marginal.
3. In some cases it may even be impossible to solve a learning problem by collecting more data. Take the compositional generalization considered in Chapter 3 and illustrated by Fig. 5.1 (copied from Fig. 1.4 in Chapter 1). The sets of hyperplanes in both subfigures identify the triangles and the blue square unambiguously, but Fig. 5.1a cannot disambiguate the red square while Fig. 5.1b can. The two models can perform perfectly on training data but not equally well on the test data, so another criteria decides between them. We expect the test data to be novel compositions of already seen attributes, so we use that to design a corresponding inductive bias.



**Fig. 5.1.:** As described in Fig. 1.4, Fig. 5.1a represents objects in a one hot fashion and Fig. 5.1b represents objects as compositions of attributes. In both figures the training data (the triangles and the blue square) can be perfectly distinguished from one another, but they do not generalize equally well to the test data (red square). An inductive bias like the population dynamics of Chapter 3 is needed to pick representations like the one in Fig. 5.1b.

This last point is echoed by results like the no free lunch theorem for machine learning [Wol96]: without any assumptions about the world or the learning problem, all approaches to learning from data are equally good on average. Superior models – *e.g.*, convolutional neural nets trained on ImageNet (as in Chapter 2) – are only superior because of the structure of the world (natural images) and the problems we choose to solve within it. Inductive biases encode more assumptions about the world into our models, and thus allow even better solutions to learning problems.

The world we design agents for implies certain inductive biases which we then encode into our model to increase its performance in that world. Current research projects focus on one model, measuring its performance in the world and finding inductive biases that improve the model. Engineers find these inductive biases by leveraging intuitions they have built up by observing the world, and how the model and its components have performed on the current task and on related tasks. This is the art of building intelligent systems. Part of what makes this an art is that the connections between performance in the world to inductive biases, and then changes to the model are both intuitive.

Instead, if we studied inductive biases more directly, then these connections might be made more scientifically. Given an inductive bias, we could try to measure aspects of the world and task to which it applies. Given an inductive bias, we could try to find many ways to implement it in a variety of models. In the long term this understanding would help us solve new tasks more quickly by decoupling these two parts of the design process. When we want to create a new model for a new task we could first predict which inductive biases would be useful, and then using a different mechanism we could predict how best to implement that inductive bias in a model. Having studied inductive biases instead of models instead of specific prediction steps for specific tasks, each of

these predictions might be more robust, relying less on the artful intuition of the engineer.

For example, take a relatively well understood inductive bias: convolution. Almost all the time convolutional layers are necessary for neural networks to achieve good performance on tasks involving natural images and we think this corresponds to the local spatial invariance of natural images. We could design metrics to try to measure properties like this – to measure the “naturalness” of images – and judge the metrics based on their ability to predict whether a convolution bias will make a big difference (comparing the best models) for a particular task. Non-convolutional and convolutional techniques work very well on classification of MNIST digits, so a metric might rate convolution as relatively unimportant for that task while it rates convolution as very important for classification of ImageNet images. Given such a metric we might be able to predict the importance of the convolution bias for images from a new domain, and thus know whether or not to include it in our model designs.

Convolution is relatively well established, but that’s because we have tested it over many datasets (different perspectives on the world) and many models. If we could do that for other properties of the world then maybe we could predict what inductive biases to use or not to use for each new task we see. A robust implementation of predictors like this might reduce the experience and intuition needed to design models, making it easier to design new models and maybe also making it easier to scale existing models to more complex problems.

## 5.1.2 Dual Process Theory, Deep Learning, and the Trajectory of AI

Dual process theory has categorized human thinking into system 1 thinking and system 2 thinking [Kah11]. System 1 uses intuition that is fast, automatic, parallel, unconscious, familiar, and bottom up. It is used to recognize objects, localize sounds, and even shortcut common problems like " $2 + 2 = ?$ ". System 2 reasons in a slow, effortful, serial, conscious, unfamiliar, top down manner. It is used to communicate your phone number, follow logical reasoning, and solve more complex math problems like " $17 * 24 = ?$ ". These categories transfer well to AI, where both types of thinking have been present since it emerged as a field.

Initially more popular, Good Old Fashioned AI (GOFAI) focuses on system 2 thinking. GOFAI representations like frames [Min74] and scripts [SA77] might represent entities in using a set of slots filled with different kinds of values. For example, a person might have a slot called "Name" filled in with the value "John" and another slot called "Is-A" filled with the value "Human" and that person can be further understood by placing them in a script which has them follow a sequence of steps like "Go To Seat" and "Order Food". An inference engine, like a production system, can use representations like these to specify a set of rules (if-then statements) and then deduce actions to take or properties of the world.

These examples are only a slice of existing work, but approaches like these have also been used to create large scale systems that encompass a surpassing amount of knowledge or capability. Long term projects like Cyc [Len95] have tried to accumulate knowledge targeted to system 2 processing in an attempt to provide a comprehensive knowledge base on top of which to build applications.



Furthermore, a variety of cognitive architectures like SOAR [Lai12] have provided frameworks for implementing a variety of AI tasks and even for implementing embodied agents that acquire their own tasks [ML16].

While there are many kinds of representation and inference that could be categorized under GOFAI and be broadly related to system 2, for the most part they rely on symbolic semantic representations. However, the world does not present us with symbolic semantic representations, rather it presents us with raw observations like the pixels of an image which have little meaning on their own. Symbolic semantic representations are implied from these raw data only after a significant amount of processing. Hence, systems like the aforementioned SOAR rely on a perception component. When built to perceive known toy environments, perception modules can easily be manually coded, and this makes sense when the goal is to study post-perception problems. But in general, perception requires system 1 thinking.

Thus system 1 thinking and system 2 thinking sometimes depend on one another. For example, consider a problem like " $17 * 24 = ?$ " written on a sheet of paper. The writing could be perceived as a string of characters: "1", "7", " ", "\*", " ", "2", "4", " ", "=", " ", "?". Or it might be perceived as numbers and symbols: "17", "\*", "24", "=", "?". In the latter case the string might be fed directly in to an arithmetic module, but in the former case the system needs to first understand how characters are composed to create meaning (*e.g.*, how "1" followed by "7" means "17"). Some thinking can be delegated to system 1 or system 2, depending on one's perspective.

To figure out how this interface should work, these systems need to interact. System 1 needs to provide semantics that are compatible with the system 2's knowledge. While system 2 may know how to add numbers, it may not know how to parse sequences of characters into addition problems. System 2 needs

to provide top down feedback that guides the semantics system 1 learns or discovers; it could reward system 1 for perceiving addition problems instead of sequences of characters. Like the work in this thesis, this top down feedback is a form of inductive bias which guides the world's representation.

This interaction isn't a particularly new idea, but taking a large step back suggests leveraging existing approaches to do this integration rather than arranging neural networks in some simplified and perhaps naïve version of system 2. If cognitive architectures like SOAR represent system 2 and deep learning represents system 1 then we could try to build an agent that uses both. A direct implementation would look like a SOAR agent which uses deep learning to do perception. The deep learning perception module would be continually updated to support the goals the SOAR agent is trying to accomplish, making those goals easier by providing representations that support reasoning efficiently. In general, sharing expertise may benefit models by making their semantics more flexible and adaptive [Bar99].

At the beginning of "Artificial Intelligence: A Modern Approach" [RN03], AI research is categorized on two axes.

1. **human-like – rational** Some research cares more about building *human-like* machines and other research cares more about building *rational* machines whether they think like humans or not.
2. **thinking – embodied** Some research cares more about building machines that *think*, and other research cares more about *acting* effectively, only thinking to the extent it benefits action.

The system 1 - system 2 distinction could join these axes.

3. **system 1 – system 2** Some research cares about *system 1* tools that are fast, automatic, parallel, unconscious, familiar, and bottom up. Other research cares about *system 2* tools that are slow, effortful, serial, conscious, unfamiliar, and top down.

Humans are often rational and thinking is often required to act effectively. Similarly, system 1 and system 2 sometimes need to interact to most effectively solve problems. These ways of thinking about AI help provide a vision to orient research and engineering over an extended perspective, and maybe for some that vision will include experts that specialize in both system 1 and system 2.

# Appendix

## A.1 Appendix for Chapter 2

### A.1.1 Details of the bias in the MNIST experiment

Recall that in Section 2.4.1 we generate biased pairs of MNIST digits by defining

$$P(l) = 0.1 \text{ and } P(r|l) = \begin{cases} 0 & \text{if } l \in \{0, \dots, 4\} \text{ and } r \in \{0, \dots, 4\} \\ 0.2 & \text{if } l \in \{0, \dots, 4\} \text{ and } r \in \{5, \dots, 9\} \\ 0.1 & \text{if } l \in \{5, \dots, 9\} \end{cases} \quad (\text{A.1})$$

and sampling left then right digits. To show that this creates a larger bias on the right than on the left, we show there is more uncertainty about left digits given right ones than right ones given left ones. That is, we show the conditional entropy  $H(l|r)$  is greater than  $H(r|l)$ .

To compute the conditional entropies, we first derive

$$P(r) = \sum_l P(r|l)P(l) = \begin{cases} 0.05 & \text{if } r \in \{0, \dots, 4\} \\ 0.15 & \text{if } r \in \{5, \dots, 9\} \end{cases} \quad (\text{A.2})$$

and

$$P(l|r) = \frac{P(r|l)P(l)}{P(r)} = \begin{cases} 0 & \text{if } l \in \{0, \dots, 4\} \text{ and } r \in \{0, \dots, 4\} \\ \frac{2}{15} & \text{if } l \in \{0, \dots, 4\} \text{ and } r \in \{5, \dots, 9\} \\ \frac{3}{15} & \text{if } l \in \{5, \dots, 9\} \text{ and } r \in \{0, \dots, 4\} \\ \frac{1}{15} & \text{if } l \in \{5, \dots, 9\} \text{ and } r \in \{5, \dots, 9\} \end{cases}. \quad (\text{A.3})$$

Using the convention  $0 \log 0 = 0$ , we can now compute

$$H(l|r) = - \sum_r P(r) \sum_l P(l|r) \log P(l|r) \approx 2.0868 \quad (\text{A.4})$$

$$H(r|l) = - \sum_l P(l) \sum_r P(r|l) \log P(r|l) \approx 1.9560 \quad (\text{A.5})$$

$$(\text{A.6})$$

## A.2 Appendix for Chapter 3

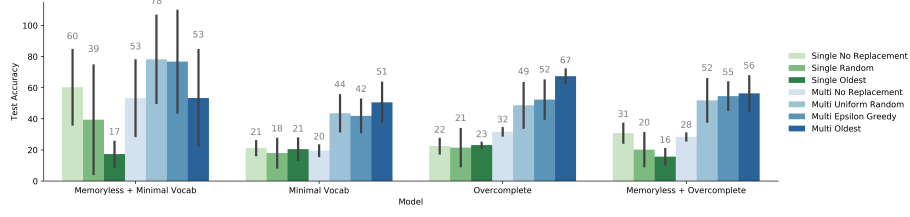
### A.2.1 Results on Novel Instance Dataset of [Kot+17]

In section 2 of the main paper we discuss the difference between the novel instance and novel pair datasets. Our novel pair dataset is a more difficult compositional split, than the one in Kottur et al. [Kot+17]. For comparison, in this section we train and evaluate our models on the novel instance from Kottur et al. [Kot+17] to show that our approach still improves compositionality in this setting and to show that our new dataset is indeed more difficult.

In (Fig. A.1) test set accuracies (with standard deviations) are reported by training and evaluating the same models as in our main results (figure 2 main paper) against the dataset from [Kot+17]. These results do not perform cross-validation, following [Kot+17]. They only vary across 4 different random seeds. Our proposed approach still outperforms models without replacement and without multiple agents. Furthermore, by comparing the approaches from (Fig. A.1) to figure 2 from the main paper we can see much lower performance across the board on the novel pair than on the novel instance dataset used here. This indicates the novel instance dataset is significantly easier than our new dataset, and that our models encourage compositionality in both settings.

### A.2.2 Replacement Strategies

Our approach to cultural transmission periodically replaces agents by re-initializing them. The approach section outlines various replacement strategies (policy  $\pi$ ), but does not detail their implementation. We do so here.



**Fig. A.1.:** Test set accuracies (with standard deviations) are reported by training and evaluating the same models as in our main results (figure 2 main paper) against the dataset from [Kot+17]. These results do not perform cross-validation, following [Kot+17]. They only vary across 4 different random seeds. See Section A.2.1.

These strategies depend on a number of possible inputs:

- $e$  the current epoch
- $E$  the period of agent replacement
- $v_i^Q/v_i^A$  the validation accuracy of agent  $i$  for Q-bots/A-bots. For Q-bots this is averaged over all potential A-bot partners, and vice-versa for A-bots.
- $a_i^Q/a_i^A$  the age in epochs of agent  $i$  for Q-bots/A-bots

Single Agent strategies are given in Algorithm 2 and Algorithm 3. Multi Agent strategies are given in Algorithm 4, Algorithm 5, and Algorithm 6. Note that Single Agent strategies always replace one agent while Multi Agent strategies always replace one Q-bot and one A-bot.

---

**Algorithm 2:** Single Agent - Random Replacement

---

```

1  $d \sim \mathcal{U}\{0, 1\}$ 
2 if  $d = 0$  then
3   return { A-bot }
4 else
5   return { Q-bot }
```

---

---

**Algorithm 3: Single Agent - Alternate Replacement**

---

```
1 Input:  $e$ 
2 if  $\lfloor e/E \rfloor = 0$  then
3   return { A-bot }
4 else
5   return { Q-bot }
```

---

---

**Algorithm 4: Multi Agent - Uniform Random Replacement**

---

```
1  $i_A \sim \mathcal{U}\{1, N_A\}$ 
2  $i_Q \sim \mathcal{U}\{1, N_Q\}$ 
3 return { A-bot  $i_A$ , Q-bot  $i_Q$  }
```

---

---

**Algorithm 5: Multi Agent - Epsilon Greedy Replacement**

---

```
1 Input:  $v_i^Q \forall i, v_i^A \forall i, \varepsilon \in [0, 1]$  (usually 0.2)
2  $d \sim \mathcal{U}[0, 1)$ 
3 if  $d < \varepsilon$  then
4    $i_A \sim \mathcal{U}\{1, N_A\}$ 
5    $i_Q \sim \mathcal{U}\{1, N_Q\}$ 
6 else
7    $i_A = \operatorname{argmin}_i v_i^A$  (unique in our experiments)
8    $i_Q = \operatorname{argmin}_i v_i^Q$  (unique in our experiments)
9 return { A-bot  $i_A$ , Q-bot  $i_Q$  }
```

---

---

**Algorithm 6: Multi Agent - Oldest Replacement**

---

```
1 Input:  $a_i^Q \forall i, a_i^V \forall i$ 
2  $i_A = \mathcal{U}\{\operatorname{argmax}_i a_i^A\}$ 
3  $i_Q = \mathcal{U}\{\operatorname{argmax}_i a_i^Q\}$ 
4 return { A-bot  $i_A$ , Q-bot  $i_Q$  }
```

---

### A.2.3 Visualization for Language Comparison at Dififerent Training Stages

In this section we visualize the language learned by agents at various stages of training to reinforce our previous conclusions and build intuition. This builds on the visualization described in section 5.3 of the main paper, so reference that section to individually understand the three sub-figures in (Fig. A.2).



Each of the three sub-figures in (Fig. A.2) summarizes all of the conversations between a particular pair of bots for the `(shape, color)` task. From left to right: (Fig. A.2a) summarizes the single pair from a Single Agent No Replacement run (3000 iterations old); (Fig. A.2b) summarizes dialogs between an old Q-bot (about 23000 iterations) and a recently re-initialized A-bot (about 3000 iterations) at the 8th and final generation of a Multi Oldest run; (Fig. A.2c) summarizes dialogs between the same old Q-bot as in (Fig. A.2b) and an old A-bot (13000 iterations) from the same Multi Oldest experiment.

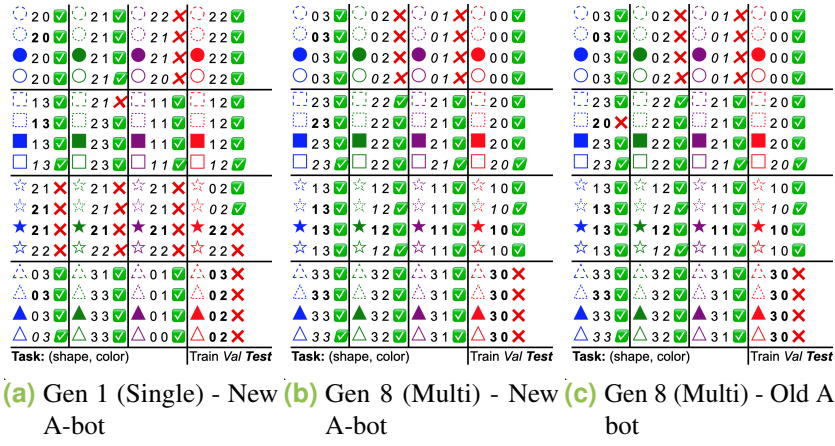
Even though the A-bots in (Fig. A.2a) and (Fig. A.2b) have trained for about<sup>1</sup> the same number of iterations, the A-bot trained in the presence of other bots which already know a functional language has already learned a somewhat compositional language whereas the Single Agent A-bot has not (Q-bot’s gets almost all star instances wrong in (Fig. A.2a), but not in (Fig. A.2b)). Furthermore, by comparing the old A-bot’s language (Fig. A.2c) with the new one (Fig. A.2b) we can see that they are extremely similar. They even lead to the same mistakes (green circles, purple circles, red triangles). This correlation in mistakes again suggests that language is transmitted between bots, in agreement with our previous experiments.

## A.2.4 Detailed Results

In our experiments we compare models and we compare replacement strategies. We ran dependent paired t-tests across random seeds, cross-val folds, and replacement strategies to compare models. We ran dependent paired t-tests across random seeds, cross-val folds, and models to compare replacement strategies. The p-values for all of these t-tests are reported here.

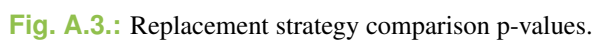
---

<sup>1</sup>Due to the stochastic nature of our Multi Agent approach.



**Fig. A.2.:** Each sub-figure summarizes an A-bot’s language, as described in section 5.3 of the main paper. By comparing the baseline of (Fig. A.2a) to a similar pair of bots from our approach (Fig. A.2b) we can see that our approach encourages compositional language to emerge. Furthermore, the similarity between (Fig. A.2b) and (Fig. A.2c) suggests language is indeed transmitted in our approach.

Replacement strategy comparisons are in (Fig. A.4) (Single Agent) and (Fig. A.5) (Multi Agent). Model comparisons are in (Fig. A.3).





**Fig. A.4.:** Single Agent model comparison p-values.



**Fig. A.5.:** Multi Agent model comparison p-values.

## A.3 Appendix for Chapter 4

### A.3.1 Architecture Details

This section describes our architecture in more detail. Algorithm 7 summarizes our complete Q-bot implementation and subsequent algorithms define the sub-routines used inside Q-bot. The planner module is described in Algorithm 9, the predictor is described in Algorithm 8, and the speaker is described in Algorithm 10. Algorithm 11 describes the encoder used for the ELBO loss.

Note that the number of bounding boxes per image is  $B$ , the number of images in a pool is  $P$ , and the max question length is  $T$ .

There are two notable differences between this section and the main paper:

- In this section there is an additional hidden state  $\bar{h}_r$  that parallels  $h_r$ . This means the query used by the context encoder in the planner to specialize, allowing  $h_r$  to focus on representing the entire dialog state. The hidden state  $h_r^{\text{main}}$  from the main paper can be thought of as a tuple of hidden states  $h_r^{\text{main}} = (h_r, \bar{h}_r)$ .
- Also note that in the main paper the interface for Q-bot includes all questions and answers from the dialog history. In our implementation we only require the previous question and answer, modeling all necessary history information through  $h_r$  and  $\bar{h}$ .

In the planner Algorithm 9 at lines 5 and 6  $g, f_1, f_3$  are all two layer MLPs with ReLU output and weight norm. Both  $f_2$  and  $f_4$  are linear transformations

---

**Algorithm 7: Q-bot**

---

```
1 Function QBot ( $\mathcal{I}, q_r, a_r, h_r, \bar{h}_r$ )
   Input:  $\mathcal{I}$ 
   Input:  $q_r$ 
   Input:  $a_r$ 
   Input:  $h_r$ 
   Input:  $\bar{h}_r$ 
   Output:  $q_{r+1}$ 
   Output:  $h_{r+1}$ 
   Output:  $\bar{h}_{r+1}$ 
   Output:  $\hat{y}_r$ 
2    $h_{r+1}, \bar{h}_{r+1}, z \leftarrow \text{Planner}(\mathcal{I}, q_r, a_r, h_r, \bar{h}_r)$ 
3    $\hat{y}_r \leftarrow \text{Predictor}(h_{r+1})$ 
4    $q_{r+1} \leftarrow \text{Speaker}(z)$ 
5   return  $q_{r+1}, h_{r+1}, \bar{h}_{r+1}, \hat{y}_r$ 
```

---

---

**Algorithm 8: Predictor**

---

```
1 Function Predictor ( $\mathcal{I}, h_{r+1}, q_0, a_0, \dots, q_r, a_r$ )
   Input:  $\mathcal{I}, I_p^b \in \mathbb{R}^{2048}$ 
   Input:  $h_{r+1}$ 
   Input:  $q_0, a_0, \dots, q_r, a_r$ 
   Output:  $\hat{y}$ 
2    $\text{Attention}(Q, K, V) = \text{softmax} g_3(g_1(Q) \odot g_2(K))V$ 
3    $f_{r+1} \leftarrow [E_q(q_{r+1}), E_a(a_{r+1})] \text{ /* fact */}$ 
4    $F \leftarrow [f_1, \dots, f_{r+1}]$ 
    $\text{/* Attention over rounds */}$ 
5    $e_F \leftarrow \text{Attention}(h_{r+1}, F, F)$ 
6    $Q_y \leftarrow [h_{r+1}, e_F]$ 
    $\text{/* Attention over bounding boxes */}$ 
7    $e_I \leftarrow \text{Attention}(Q_y, \mathbf{x}, \mathbf{x}) \in \mathbb{R}^{P \times 2048}$ 
8    $e_I \leftarrow g_1(e_I)$ 
9    $Q_p \leftarrow g_2(Q_y)$ 
10   $l_y \leftarrow g_3(Q_p \odot e_I)$ 
11   $\hat{y}_r \leftarrow \text{argmax softmax } l_y$ 
12  return  $\hat{y}_r$ 
```

---

with weight norm applied (no activation function).  $f_5$  is a linear transformation without weight norm purely for dimensionality reduction. To compute  $\bar{h}_{r+1}$  we also add new linear weights  $W_1$  and  $W_2$  as for a standard LSTM output gate.

Note that for the planner there is an additional residual connection at line 16 which augments the hidden state. This allows gradients to flow through the

---

**Algorithm 9: Planner**

---

```
1 Function Planner ( $\mathcal{I}, q_r, a_r, h_r, \bar{h}_r$ )
   Input:  $\mathcal{I}, I_p^b \in \mathbb{R}^{2048}$ 
   Input:  $q_r$ 
   Input:  $a_r$ 
   Input:  $h_r$ 
   Output:  $h_{r+1}$ 
   Output:  $\bar{h}_{r+1}$ 
   Output:  $z$ 
   /* Context Coder                                     */
2    $e_q \leftarrow E_q(q_r)$ 
3    $e_a \leftarrow E_a(a_r)$ 
4    $e_c \leftarrow f_5([\bar{h}_r, e_q, e_a])$ 
5    $\alpha_p \leftarrow \text{softmax} f_2(g(e_c) \odot f_1(I_p^b))$ 
6    $\beta_p^b \leftarrow \text{softmax} f_4(g(e_c) \odot f_3(I_p^b))$ 
7    $\hat{v}_r \leftarrow \sum_{p=1}^P \sum_{b=1}^B \alpha_p \beta_p^b I_p^b$ 
8    $x_r^{\text{context}} \leftarrow [\hat{v}_r, e_q, e_a]$ 
   /* Dialog RNN                                         */
9    $h_{r+1}, c_{r+1} \leftarrow \gamma(x_r^{\text{context}}, h_r)$ 
10   $h_{r+1} \leftarrow \text{Dropout}(h_{r+1})$ 
11   $\bar{h}_{r+1} \leftarrow \sigma(W_1^T x_r^{\text{context}} + W_2^T h_r) \odot \tanh(c_{r+1})$ 
12   $\bar{h}_{r+1} \leftarrow \text{Dropout}(\bar{h}_{r+1})$ 
   /* Question Policy                                    */
13   $h_{r+1}^z \leftarrow W_z^T h_{r+1} \in \mathbb{R}^d$ 
14   $l_{Kn:K(n+1)-1}^z \leftarrow \text{LogSoftmax}(h_{Kn:K(n+1)-1}^z) \forall n$ 
15   $z_n \leftarrow \text{GumbelSoftmax}(l_{Kn:K(n+1)-1}^z) \forall n$ 
16   $h_{r+1} \leftarrow h_{r+1} + \text{ReLU}(W_l^T l^z)$ 
17  return  $h_{r+1}, \bar{h}_{r+1}, z$ 
```

---

question policy parameters  $W_z$  at line 12 when we fine-tune for task performance without fully supervised dialogs.

In Algorithm 8  $g_1, g_2$  are both 2-layer ReLU nets with weight norm. Also  $g_3$  is a 2-layer net with ReLU and Dropout on the hidden activation and weight norm on both layers.

In Algorithm 10  $\beta$  is an LSTM decoder.



---

**Algorithm 10: Speaker**

---

```
1 Function Speaker ( $z$ )  
   Input:  $z$   
   Output:  $q_{r+1}$   
2    $e_z \leftarrow \sum_{n=0}^{N-1} E_n^z(z_n)$   
3    $q_{r+1} \leftarrow \beta(e_z)$   
4   return  $q_{r+1}$ 
```

---

---

**Algorithm 11: Encoder**

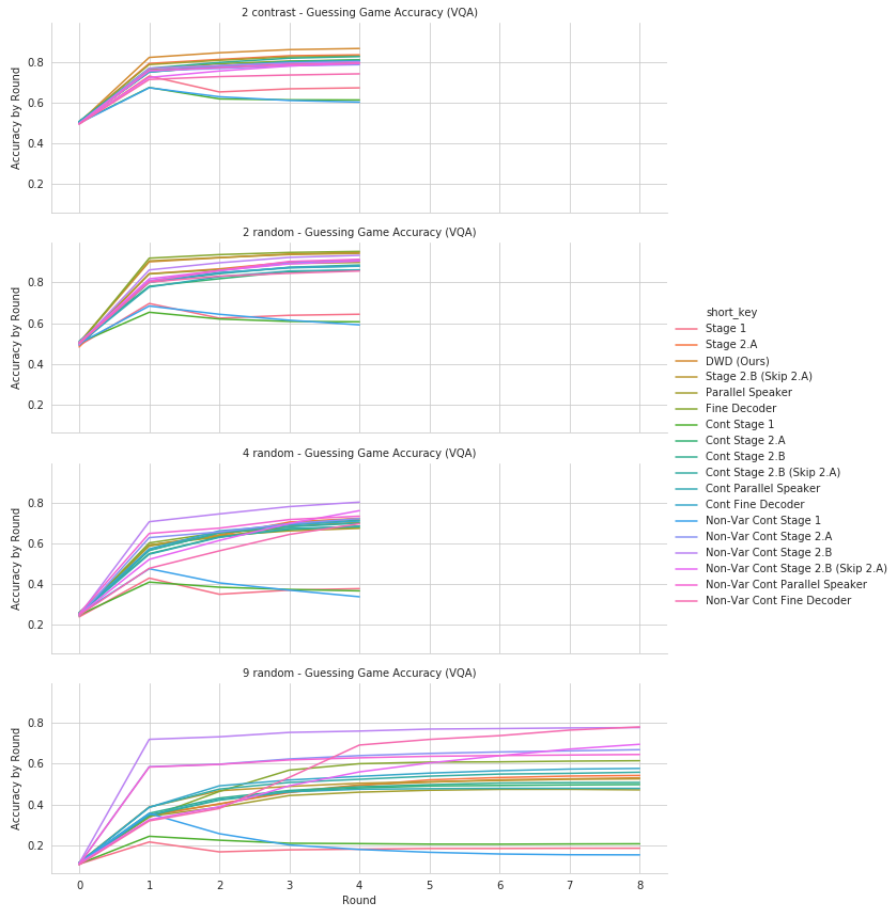
---

```
1 Function Encoder ( $\mathcal{I}, q_r$ )  
   Input:  $\mathcal{I}, I_p^b \in \mathbb{R}^{2048}$   
   Input:  $q_r$   
   Output:  $z$  (sample or distribution parameters)  
   /* Context Coder */  
2    $e_q \leftarrow E_q(q_r)$   
3    $\alpha_p \leftarrow \text{softmax} f_2(g(e_q) \odot f_1(I_p^b))$   
4    $\beta_p^b \leftarrow \text{softmax} f_4(g(e_q) \odot f_3(I_p^b))$   
5    $\hat{v} \leftarrow \sum_{p=1}^P \sum_{b=1}^B \alpha_p \beta_p^b I_p^b$   
6    $h^z \leftarrow W_z^T \hat{v}$   
7    $l_{Kn:K(n+1)-1}^z \leftarrow \text{LogSoftmax}(h_{Kn:K(n+1)-1}^z) \forall n$   
8    $z_n \leftarrow \text{GumbelSoftmax}(l_{Kn:K(n+1)-1}^z) \forall n$   
9   return  $z$ 
```

---

### A.3.2 Additional Results

Experiments in the main paper considered dialog performance after the first round (top of Table 1) and at the final round of dialog (either 5 or 9 depending on pool size). This does not give much sense for how dialog performance increases over rounds of dialog, so we report Q-bot’s guessing game performance at each round of dialog in (Fig. A.6). For all fine-tuned models performance goes up over multiple rounds of dialog, though some models benefit more than others. Stage 1 models decrease in performance after round 1 because it is too far from the training data such models have been exposed to.



**Fig. A.6.:** Task performance (guessing game accuracy) over rounds of dialog. Performance increases over rounds for all models except the Stage 1 models.

### A.3.3 Mechanical Turk Studies

In the experiments section we described two studies where we asked humans to compare questions.

In the relevance study turkers were presented with the interface depicted in (Fig. A.7). It asked them to compare questions based on their relevance to any image in the image pool. The question with higher relevance should have been picked even if the question was not very grammatical. All model pairs were evaluated for each pool of images. The questions were presented in a random order, though the Equally relevant option was always last.

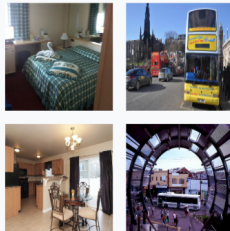
## Please help us compare questions!

Show / Hide Instructions

### Instructions

Which question is most relevant to the images?

A question is relevant to the images if it accurately refers to the contents of at least one of the 4 images presented next to the question. If it does not refer to any image content then it is not relevant. Note that how relevant a question is does NOT depend on how grammatical or fluent it is. In particular, note that nonsense questions may still be relevant if they accurately identify image contents.



Consider this image pool. Here are some examples of questions that are relevant or irrelevant to the pool:

- The question "What is on the bed?" is MORE relevant than "What is in the image?" because it refers to specific content in one of the images.
- The question "Is the bike red?" is NOT relevant because there is not an image with a bike. It would be relevant if there were a bike.
- "Is the bike red?" is LESS relevant than a non-grammatical question that mentions specific image content, like "What is the yellow bus sitting?"
- "Is the bike red?" and "Remain is going?" are EQUALLY (ir)relevant.

For each pool of images shown below, click on the **most relevant** question.



The  
question

- ☐ Slowly fresh is is a ring?
- ☒ Is this an overcast day?
- ☐ (Equally relevant)

is more relevant.

**Fig. A.7.:** An example of the interface used to ask humans to evaluate question relevance.

Please help us compare questions!

Show / Hide Instructions

Instructions

Which question is more understandable?

For example, a question may be less understandable if it

- has bad grammar and is not fluent English, like "man doing is what?" and "language spelled is this vehicle?"
- is using words that don't make sense in context, like "what decor value is shown in focus?"
- etc.

For each pair of questions shown below, click on the one that is **more understandable**. Mark Equally understandable if both questions make sense or if both questions are not understandable.

The question

☐
Protruding what number is on this?

☐
How many people are under the snow on the ground?

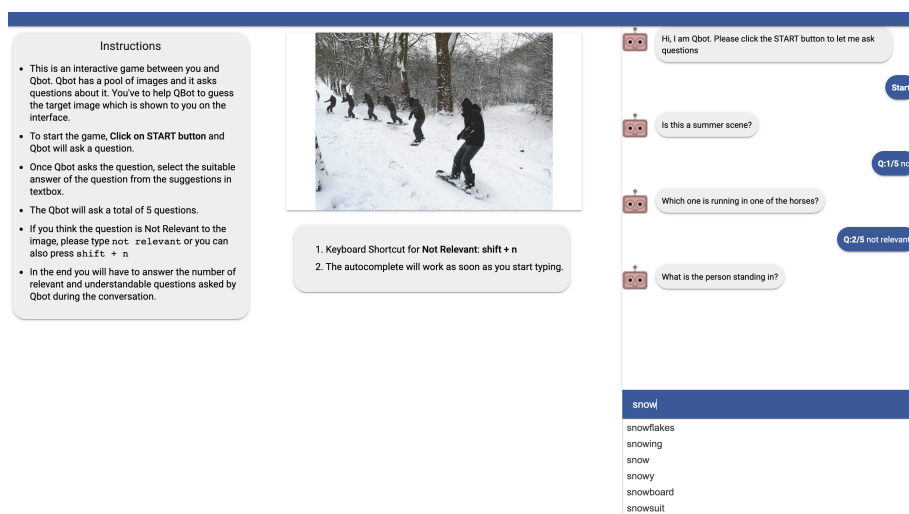
☒
(Equally understandable)

is more understandable.

**Fig. A.8.:** An example of the interface used to ask humans to evaluate question fluency.

In the fluency study ((Fig. A.8)) turkers were presented with the same pairs of questions as in the relevance interface but they were not given image pools with which to associate the questions. We asked them to compare questions based on how well they could be understood. As in the relevance study questions were presented in a random order.

In the figure 4, we display the interface which was used to pair up the Q-bot with a human in real time. The Q-bot asks a question in order to guess the target image and a human answers the question by looking at the target image. This sequence of question/answer starts with a random guess from Q-bot and goes on for 4 Rounds.



**Fig. A.9.:** An example of the interface subjects used to interact with our Q-bot models.

# Bibliography

- [ADK17] Jacob Andreas, Anca D. Dragan, and Dan Klein. „Translating Neuralese“. In: *ACL*. 2017 (cit. on p. 38).
- [And+13] Galen Andrew, Raman Arora, Jeff Bilmes, and Karen Livescu. „Deep canonical correlation analysis“. In: *Proceedings of the 30th International Conference on Machine Learning*. 2013, pp. 1247–1255 (cit. on pp. 23, 24).
- [And+18] Peter Anderson, Xiaodong He, Chris Buehler, et al. „Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering“. In: *CVPR*. 2018 (cit. on pp. 1, 61).
- [Ant+15] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, et al. „VQA: Visual Question Answering“. In: *International Conference on Computer Vision (ICCV)*. 2015 (cit. on pp. 1, 2, 13, 17).
- [Ash+18] Ashwin K Vijayakumar, Michael Cogswell, Ramprasath R. Selvaraju, et al. „Diverse Beam Search: Decoding Diverse Solutions from Neural Sequence Models“. In: *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI)*. 2018 (cit. on p. 69).
- [Bar61] Horace B Barlow. „Possible principles underlying the transformations of sensory messages“. In: (1961) (cit. on p. 22).
- [Bar99] Lawrence W. Barsalou. „Perceptual symbol systems.“ In: *The Behavioral and brain sciences* 22 4 (1999), 577–609; discussion 610–60 (cit. on p. 87).
- [BB09] Yoshua Bengio and James S Bergstra. „Slow, decorrelated features for pre-training complex cell-like networks“. In: *Advances in neural information processing systems*. 2009, pp. 99–107 (cit. on p. 23).
- [BCV13] Yoshua Bengio, Aaron C. Courville, and Pascal Vincent. „Representation Learning: A Review and New Perspectives“. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35 (2013), pp. 1798–1828 (cit. on pp. 3, 6).
- [Ben12] Yoshua Bengio. „Evolving Culture vs Local Minima“. In: *CoRR* abs/1203.2990 (2012) (cit. on p. 54).

- [Bre96] Leo Breiman. „Bagging predictors“. In: *Machine learning* 24.2 (1996), pp. 123–140 (cit. on p. 19).
- [Bri00] Ted Briscoe. „Grammatical Acquisition: Inductive Bias and Coevolution of Language and the Language Acquisition Device“. In: *Language*. Vol. 76. Linguistic Society of America, 2000 (cit. on p. 52).
- [Bri02] Henry Brighton. „Compositional Syntax From Cultural Transmission“. In: *Artificial Life* 8 (2002), pp. 25–54 (cit. on p. 52).
- [Ceg16] Maciej Ceglowski. *Superintelligence - The Idea That Eats Smart People*. 2016 (cit. on p. 80).
- [Cha+15] Sarath Chandar, Mitesh M Khapra, Hugo Larochelle, and Balaraman Ravindran. „Correlational Neural Networks“. In: *arXiv preprint arXiv:1504.07225* (2015) (cit. on pp. 23, 24).
- [Cha+17] Prithvijit Chattopadhyay, Deshraj Yadav, Viraj Prabhu, et al. „Evaluating Visual Conversational Agents via Cooperative Human-AI Games“. In: *Proceedings of the Fifth AAAI Conference on Human Computation and Crowdsourcing (HCOMP)*. 2017 (cit. on pp. 74, 79).
- [Che+14] Brian Cheung, Jesse A Livezey, Arjun K Bansal, and Bruno A Olshausen. „Discovering Hidden Factors of Variation in Deep Networks“. In: *arXiv preprint arXiv:1412.6583* (2014) (cit. on p. 24).
- [Che+16] Xi Chen, Yan Duan, Rein Houthoofd, et al. „InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets“. In: *NIPS*. 2016 (cit. on p. 6).
- [CK03] Morten H. Christiansen and Simon Kirby. „Language evolution: consensus and controversies.“ In: *Trends in cognitive sciences* 7 7 (2003), pp. 300–307 (cit. on p. 53).
- [CLF18] Edward Choi, Angeliki Lazaridou, and Nando de Freitas. „Compositional Obverter Communication Learning From Raw Visual Input“. In: *International Conference on Learning Representations (ICLR)*. 2018 (cit. on p. 53).
- [CTK09] Hannah Cornish, Monica Tamariz, and Simon Kirby. „Complex Adaptive Systems and the Origins of Adaptive Structure: What Experiments Can Tell Us“. In: *Language Learning* 59.s1 (2009), pp. 187–205. eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-9922.2009.00540.x> (cit. on p. 53).
- [CZ15] Xinlei Chen and C Lawrence Zitnick. „Mind’s eye: A recurrent visual representation for image caption generation“. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 2422–2431 (cit. on p. 17).
- [Das+17a] Abhishek Das, Satwik Kottur, José M.F. Moura, Stefan Lee, and Dhruv Batra. „Learning Cooperative Visual Dialog Agents with Deep Reinforcement Learning“. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2017 (cit. on pp. 38, 56, 57, 60, 77, 78).

- [Das+17b] Abhishek Das, Satwik Kottur, Khushi Gupta, et al. „Visual Dialog“. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), pp. 1080–1089 (cit. on p. 3).
- [DB17] Emily L. Denton and Vighnesh Birodkar. „Unsupervised Learning of Disentangled Representations from Video“. In: *NIPS*. 2017 (cit. on p. 6).
- [DHB19] Gautier Dagan, Dieuwke Hupkes, and Elia Bruni. „Co-evolution of language and agents in referential games“. In: *pre-print* (2019) (cit. on p. 54).
- [DK17] Samuel F. Dodge and Lina J. Karam. „A Study and Comparison of Human and Deep Learning Recognition Performance under Visual Distortions“. In: *2017 26th International Conference on Computer Communication and Networks (ICCCN)* (2017), pp. 1–7 (cit. on p. 1).
- [Don+14] Jeff Donahue, Yangqing Jia, Oriol Vinyals, et al. „DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition.“ In: *Proceedings of the International Conference on Machine Learning (ICML)*. 2014 (cit. on p. 18).
- [Esm+18] Babak Esmaeili, Hao Wu, Sarthak Jain, et al. „Structured Disentangled Representations“. In: *AISTATS*. 2018 (cit. on p. 6).
- [Foe+16] Jakob Foerster, Ioannis Alexandros Assael, Nando de Freitas, and Shimon Whiteson. „Learning to communicate with deep multi-agent reinforcement learning“. In: *Advances in Neural Information Processing Systems*. 2016, pp. 2137–2145 (cit. on pp. 38, 53).
- [GB10] Xavier Glorot and Yoshua Bengio. „Understanding the difficulty of training deep feedforward neural networks“. In: *International conference on artificial intelligence and statistics*. 2010, pp. 249–256 (cit. on p. 28).
- [GB16] Çağlar Gülçehre and Yoshua Bengio. „Knowledge Matters: Importance of Prior Information for Optimization“. In: *Journal of Machine Learning Research* 17 (2016), 8:1–8:32 (cit. on p. 54).
- [GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. <http://www.deeplearningbook.org>. MIT Press, 2016 (cit. on p. 6).
- [Gir+14] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. „Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation“. In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2014 (cit. on p. 18).
- [Gir15] Ross Girshick. „Fast R-CNN“. In: *arXiv preprint arXiv:1504.08083* (2015) (cit. on p. 35).
- [Goo+13] Ian J Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. „Maxout networks“. In: *arXiv preprint arXiv:1302.4389* (2013) (cit. on p. 18).



- [Goy+17] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. „Making the V in VQA Matter: Elevating the Role of Image Understanding in Visual Question Answering“. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017 (cit. on pp. 64, 78).
- [He+16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. „Deep Residual Learning for Image Recognition“. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 770–778 (cit. on p. 1).
- [Hig+17] Irina Higgins, Loïc Matthey, Arka Pal, et al. „beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework“. In: *ICLR*. 2017 (cit. on pp. 6, 8).
- [HS90] Lars Kai Hansen and Peter Salamon. „Neural network ensembles“. In: *IEEE transactions on pattern analysis and machine intelligence* 12.10 (1990), pp. 993–1001 (cit. on p. 19).
- [HT17] Serhii Havrylov and Ivan Titov. „Emergence of Language with Multi-agent Games: Learning to Communicate with Sequences of Symbols“. In: *NIPS*. 2017 (cit. on p. 53).
- [Hu+19] Hengyuan Hu, Denis Yarats, Qucheng Gong, Yuandong Tian, and Mike Lewis. „Hierarchical decision making by generating and following natural language instructions“. In: *arXiv preprint arXiv:1906.00744* (2019) (cit. on p. 78).
- [IS15] Sergey Ioffe and Christian Szegedy. „Batch normalization: Accelerating deep network training by reducing internal covariate shift“. In: *arXiv preprint arXiv:1502.03167* (2015) (cit. on p. 24).
- [JGP17] Eric Jang, Shixiang Gu, and Ben Poole. „Categorical Reparametrization with Gumbel-Softmax“. In: *International Conference on Learning Representations (ICLR)*. 2017 (cit. on pp. 67, 77).
- [Jia13] Yangqing Jia. *Caffe: An Open Source Convolutional Architecture for Fast Feature Embedding*. <http://caffe.berkeleyvision.org/>. 2013 (cit. on p. 26).
- [Jia+18] Yu Jiang, Vivek Natarajan, Xinlei Chen, et al. „Pythia v0.1: the Winning Entry to the VQA Challenge 2018“. In: *CoRR* abs/1807.09956 (2018) (cit. on pp. 1, 2).
- [Kah11] Daniel Kahneman. *Thinking, fast and slow*. New York: Farrar, Straus and Giroux, 2011 (cit. on p. 85).
- [Kai+18] Lukasz Kaiser, Aurko Roy, Ashish Vaswani, et al. „Fast Decoding in Sequence Models Using Discrete Latent Variables“. In: *ICML*. 2018 (cit. on p. 67).
- [Kan+19] Dongyeop Kang, Anusha Balakrishnan, Pararth Shah, et al. „Recommendation as a Communication Game: Self-Supervised Bot-Play for Goal-oriented Dialogue“. In: *arXiv preprint arXiv:1909.03922* (2019) (cit. on p. 78).

- [KB15] Diederik P. Kingma and Jimmy Ba. „Adam: A Method for Stochastic Optimization“. In: *International Conference on Learning Representations (ICLR)*. 2015 (cit. on p. 45).
- [KCS08] Simon Kirby, Hannah Cornish, and Kenny Smith. „Cumulative cultural evolution in the laboratory: An experimental approach to the origins of structure in human language“. In: *Proceedings of the National Academy of Sciences* (2008). eprint: <https://www.pnas.org/content/early/2008/07/29/0707835105.full.pdf> (cit. on pp. 11, 37, 38, 43, 53).
- [KGS14] Simon Kirby, Tom Griffiths, and Kenny Smith. „Iterated learning and the evolution of language“. In: *Current Opinion in Neurobiology* 28 (2014), pp. 108–114 (cit. on pp. 11, 38, 52, 53).
- [KH09] Alex Krizhevsky and Geoffrey Hinton. „Learning multiple layers of features from tiny images“. In: *Computer Science Department, University of Toronto, Tech. Rep* 1.4 (2009), p. 7 (cit. on pp. 18, 20, 29, 31).
- [Kir01] Simon Kirby. „Spontaneous evolution of linguistic structure-an iterated learning model of the emergence of regularity and irregularity“. In: *IEEE Trans. Evolutionary Computation* 5 (2001), pp. 102–110 (cit. on pp. 11, 37–39, 43, 53).
- [Kir02] Simon Kirby. „Natural Language From Artificial Life“. In: *Artificial Life*. 2002 (cit. on p. 53).
- [Kot+17] Satwik Kottur, José M. F. Moura, Stefan Lee, and Dhruv Batra. „Natural Language Does Not Emerge ‘Naturally’ in Multi-Agent Dialog“. In: *EMNLP*. 2017 (cit. on pp. 38–42, 44–46, 48, 53, 91, 92).
- [KSB17] Abhishek Kumar, Prasanna Sattigeri, and Avinash Balakrishnan. „Variational Inference of Disentangled Latent Concepts from Unlabeled Observations“. In: *ICLR* (2017) (cit. on p. 8).
- [KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoff Hinton. „ImageNet Classification with Deep Convolutional Neural Networks“. In: *NIPS*. 2012 (cit. on pp. 1, 17, 20, 31).
- [Kul+15] Tejas D. Kulkarni, William F. Whitney, Pushmeet Kohli, and Joshua B. Tenenbaum. „Deep Convolutional Inverse Graphics Network“. In: *ArXiv abs/1503.03167* (2015) (cit. on p. 6).
- [Lai12] John E. Laird. „The Soar Cognitive Architecture“. In: 2012 (cit. on p. 86).
- [Laz+18] Angeliki Lazaridou, Karl Moritz Hermann, Karl Tuyls, and Stephen Clark. „Emergence of Linguistic Communication from Referential Games with Symbolic and Pixel Input“. In: *International Conference on Learning Representations (ICLR)*. 2018 (cit. on p. 53).
- [LB18] Brenden M. Lake and Marco Baroni. „Generalization without Systematicity: On the Compositional Skills of Sequence-to-Sequence Recurrent Networks“. In: *ICML*. 2018 (cit. on p. 37).

- [LB19] Fushan Li and Michael Bowling. „Ease-of-Teaching and Language Structure from Emergent Communication“. In: *CoRR* abs/1906.02403 (2019). arXiv: 1906.02403 (cit. on p. 54).
- [LBH15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. „Deep learning“. In: *nature* 521.7553 (2015), p. 436 (cit. on p. 6).
- [LCY13] Min Lin, Qiang Chen, and Shuicheng Yan. „Network in network“. In: *arXiv preprint arXiv:1312.4400* (2013) (cit. on pp. 20, 34).
- [LeC+95] Yann LeCun, LD Jackel, L Bottou, et al. „Comparison of learning algorithms for handwritten digit recognition“. In: *International conference on artificial neural networks*. Vol. 60. 1995, pp. 53–60 (cit. on pp. 20, 26).
- [Lee+18] Jason D. Lee, Kyunghyun Cho, Jason Weston, and Douwe Kiela. „Emergent Translation in Multi-Agent Communication“. In: *CoRR* abs/1710.06922 (2018) (cit. on p. 53).
- [Len95] Douglas B. Lenat. „CYC: a large-scale investment in knowledge infrastructure“. In: *Commun. ACM* 38 (1995), pp. 32–38 (cit. on p. 85).
- [Lew+17] Mike Lewis, Denis Yarats, Yann Dauphin, Devi Parikh, and Dhruv Batra. „Deal or No Deal? End-to-End Learning for Negotiation Dialogues“. In: *EMNLP*. 2017 (cit. on pp. 56, 57, 78).
- [Lew69] David Lewis. „Convention: A Philosophical Study“. In: 1969 (cit. on p. 79).
- [Li+15] Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and William B. Dolan. „A Diversity-Promoting Objective Function for Neural Conversation Models“. In: *HLT-NAACL*. 2015 (cit. on p. 69).
- [Lin+14a] Tsung-Yi Lin, Michael Maire, Serge Belongie, et al. *Microsoft COCO: Common Objects in Context*. 2014. eprint: arXiv:1405.0312 (cit. on p. 17).
- [Lin+14b] T.Y. Lin, M. Maire, S. Belongie, et al. „Microsoft COCO: Common Objects in Context“. In: *Proceedings of European Conference on Computer Vision (ECCV)*. 2014 (cit. on p. 67).
- [Lin88] Ralph Linsker. „Self-organization in a perceptual network“. In: *Computer* 21.3 (1988), pp. 105–117 (cit. on p. 22).
- [Loc+19] Francesco Locatello, Stefan Bauer, Mario Lucic, et al. „Challenging Common Assumptions in the Unsupervised Learning of Disentangled Representations“. In: *ICML*. 2019 (cit. on p. 6).
- [LPB17] Angeliki Lazaridou, Alexander Peysakhovich, and Marco Baroni. „Multi-Agent Cooperation and the Emergence of (Natural) Language“. In: *International Conference on Learning Representations (ICLR)*. 2017 (cit. on pp. 38, 53).
- [Lu+17] Jiasen Lu, Anitha Kannan, Jianwei Yang, Devi Parikh, and Dhruv Batra. „Best of Both Worlds: Transferring Knowledge from Discriminative Learning to a Generative Visual Dialog Model“. In: *NIPS*. 2017 (cit. on p. 77).

- [Lu+19] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. „ViLBERT: Pre-training Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks“. In: *ArXiv abs/1908.02265* (2019) (cit. on p. 1).
- [MA18] Igor Mordatch and Pieter Abbeel. „Emergence of Grounded Compositional Language in Multi-Agent Populations“. In: *AAAI*. 2018 (cit. on pp. 39, 48, 53).
- [Mil+17] A. H. Miller, W. Feng, A. Fisch, et al. „ParLAI: A Dialog Research Software Platform“. In: *arXiv preprint arXiv:1705.06476* (2017) (cit. on p. 56).
- [Min74] Marvin Minsky. „A framework for representing knowledge“. In: 1974 (cit. on p. 85).
- [Mir+17] Mircea Mironenco, Dana Kianfar, Ke Tran, Evangelos Kanoulas, and Efstratios Gavves. „Examining cooperation in visual dialog models“. In: *arXiv preprint arXiv:1712.01329* (2017) (cit. on p. 60).
- [Mis+18] Ishan Misra, Ross B. Girshick, Rob Fergus, et al. „Learning by Asking Questions“. In: *CVPR* (2018) (cit. on p. 78).
- [ML16] Aaron Mininger and John Laird. „Interactively Learning Strategies for Handling References to Unseen or Unknown Objects“. In: 2016 (cit. on p. 86).
- [MMT17] Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. „The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables“. In: *International Conference on Learning Representations (ICLR)*. 2017 (cit. on p. 65).
- [Mni+15] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, et al. „Human-level control through deep reinforcement learning“. In: *Nature* 518 (2015), pp. 529–533 (cit. on p. 1).
- [Mon+14] Guido Montúfar, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio. „On the Number of Linear Regions of Deep Neural Networks“. In: *ArXiv abs/1402.1869* (2014) (cit. on p. 9).
- [Mos+16] Nasrin Mostafazadeh, Ishan Misra, Jacob Devlin, et al. „Generating Natural Questions About an Image“. In: *ACL* (2016) (cit. on p. 14).
- [NK99] Martin A. Nowak and David C. Krakauer. „The evolution of language.“ In: *Proceedings of the National Academy of Sciences of the United States of America* 96 14 (1999), pp. 8028–33 (cit. on p. 52).
- [NPJ00] Martin A. Nowak, Joshua B. Plotkin, and Vincent A A Jansen. „The evolution of syntactic communication“. In: *Nature* 404 (2000), pp. 495–498 (cit. on pp. 48, 52, 53).
- [PB90] Steven Pinker and Paul Bloom. „Natural language and natural selection“. In: *Behavioral and brain sciences* 13.4 (1990), pp. 707–727 (cit. on p. 52).
- [PC93] Michael P. Perrone and Leao N. Cooper. „When Networks Disagree: Ensemble Methods for Hybrid Neural Networks“. In: *Tech Report*. Chapman and Hall, 1993, pp. 126–142 (cit. on p. 19).

- [Per02] Amy Perfors. „Simulated Evolution of Language: a Review of the Field“. In: *J. Artificial Societies and Social Simulation* 5 (2002) (cit. on p. 52).
- [PH86] Barak A Pearlmutter and Geoffrey Hinton. „G-maximization: An unsupervised learning procedure for discovering regularities“. In: *AIP conference proceedings*. Vol. 151. American Institute of Physics. 1986, pp. 333–338 (cit. on p. 23).
- [RMLA18] Limor Raviv, Antje Meyer, and Shiri Lev-Ari. „Compositional structure can emerge without generational transmission“. In: *Cognition* 182 (2018), pp. 151–164 (cit. on pp. 44, 47, 53).
- [RN03] Stuart J. Russell and Peter Norvig. „Artificial intelligence - a modern approach, 2nd Edition“. In: *Prentice Hall series in artificial intelligence*. 2003 (cit. on p. 87).
- [Rus+14] Olga Russakovsky, Jia Deng, Hao Su, et al. „Imagenet large scale visual recognition challenge“. In: *International Journal of Computer Vision* (2014), pp. 1–42 (cit. on pp. 1, 2, 17, 20).
- [SA77] Roger C. Schank and Robert P. Abelson. „Scripts, Plans, Goals And Understanding: An Inquiry Into Human Knowledge Structures“. In: 1977 (cit. on p. 85).
- [Sch92] Jürgen Schmidhuber. „Learning factorial codes by predictability minimization“. In: *Neural Computation* 4.6 (1992), pp. 863–879 (cit. on p. 23).
- [Ser+16] Iulian V Serban, II Ororbia, G Alexander, Joelle Pineau, and Aaron Courville. „Piecewise latent variables for neural variational text processing“. In: *arXiv preprint arXiv:1612.00377* (2016) (cit. on p. 78).
- [Ser+17] Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, et al. „A hierarchical latent variable encoder-decoder model for generating dialogues“. In: *Thirty-First AAAI Conference on Artificial Intelligence*. 2017 (cit. on p. 78).
- [Sil+16] David Silver, Aja Huang, Christopher J. Maddison, et al. „Mastering the game of Go with deep neural networks and tree search“. In: *Nature* 529 (2016), pp. 484–503 (cit. on p. 1).
- [Sil+17] David Silver, Thomas Hubert, Julian Schrittwieser, et al. „Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm“. In: *ArXiv abs/1712.01815* (2017) (cit. on p. 1).
- [SKB03] Kenny Smith, Simon Kirby, and Henry Brighton. „Iterated Learning: A Framework for the Emergence of Language“. In: *Artificial Life* 9 (2003), pp. 371–386 (cit. on pp. 52, 53).
- [Smi06] Kenny Smith. „Cultural Evolution of Language“. In: *Encyclopedia of Language and Linguistics 2 Edition* 2 (2006), pp. 315–322 (cit. on p. 52).
- [Spi+17] Matthew Spike, Kevin Stadler, Simon Kirby, and Kenny Smith. „Minimal Requirements for the Emergence of Learned Signaling“. In: *Cognitive Science*. 2017 (cit. on p. 52).

- [SPK10] Thomas C. Scott-Phillips and Simon Kirby. „Language evolution in the laboratory.“ In: *Trends in cognitive sciences* 14 9 (2010), pp. 411–7 (cit. on p. 53).
- [Sri+14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. „Dropout: A simple way to prevent neural networks from overfitting“. In: *The Journal of Machine Learning Research* 15.1 (2014), pp. 1929–1958 (cit. on pp. 18, 24, 25, 27, 28).
- [SSF16] Sainbayar Sukhbaatar, Arthur Szlam, and Rob Fergus. „Learning Multi-agent Communication with Backpropagation“. In: *NIPS*. 2016 (cit. on pp. 38, 53).
- [Ste+19] Sjoerd van Steenkiste, Francesco Locatello, Jurgen Schmidhuber, and Olivier Bachem. „Are Disentangled Representations Helpful for Abstract Visual Reasoning?“ In: *ArXiv abs/1905.12506* (2019) (cit. on p. 6).
- [SZ14] Karen Simonyan and Andrew Zisserman. „Very Deep Convolutional Networks for Large-Scale Image Recognition“. In: *ICLR* (2014) (cit. on p. 1).
- [Ten+17] Damien Teney, Peter Anderson, Xiaodong He, and Anton van den Hengel. „Tips and Tricks for Visual Question Answering: Learnings from the 2017 Challenge“. In: *CoRR abs/1708.02711* (2017) (cit. on p. 60).
- [Tib96] Robert Tibshirani. „Regression Shrinkage and Selection via the Lasso“. In: *Journal of the Royal Statistical Society, Series B* 58 (1 1996), pp. 267 –288 (cit. on p. 18).
- [Tik43] Andrey Nikolayevich Tikhonov. „On the stability of inverse problems“. In: *Dokl. Akad. Nauk SSSR* (1943), pp. 195–198 (cit. on p. 18).
- [TOB15] Lucas Theis, Aäron van den Oord, and Matthias Bethge. „A note on the evaluation of generative models“. In: *CoRR abs/1511.01844* (2015) (cit. on p. 69).
- [Tom99] Michael Tomasello. *The cultural origins of human cognition*. Harvard university press, 1999 (cit. on p. 52).
- [Vin+14] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. „Show and Tell: A Neural Image Caption Generator“. In: (2014). *arXiv: 1411.4555* (cit. on p. 17).
- [Vog05] Paul Vogt. „The emergence of compositional structures in perceptually grounded language games“. In: *Artificial intelligence* 167.1-2 (2005), pp. 206–242 (cit. on p. 52).
- [Vri+16] Harm de Vries, Florian Strub, A. P. Sarath Chandar, et al. „GuessWhat?! Visual Object Discovery through Multi-modal Dialogue“. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 4466–4475 (cit. on p. 79).
- [Wah+11] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. *The Caltech-UCSD Birds-200-2011 Dataset*. Tech. rep. CNS-TR-2011-001. California Institute of Technology, 2011 (cit. on p. 68).

- [Wan+13] Li Wan, Matthew Zeiler, Sixin Zhang, Yann L. Cun, and Rob Fergus. „Regularization of neural networks using dropconnect“. In: *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*. 2013, pp. 1058–1066 (cit. on p. 18).
- [WAZ17] Jason D. Williams, Kavosh Asadi, and Geoffrey Zweig. „Hybrid code networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning“. In: *arXiv preprint arXiv:1702.03274* (2017) (cit. on p. 78).
- [Wen+17] Tsung-Hsien Wen, Yishu Miao, Phil Blunsom, and Steve Young. „Latent intention dialogue models“. In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 3732–3741 (cit. on p. 78).
- [Wol96] David H. Wolpert. „The Lack of A Priori Distinctions Between Learning Algorithms“. In: *Neural Computation* 8 (1996), pp. 1341–1390 (cit. on p. 83).
- [Xia+18] Yongqin Xian, Christoph H. Lampert, Bernt Schiele, and Zeynep Akata. „Zero-shot learning-a comprehensive evaluation of the good, the bad and the ugly“. In: *IEEE transactions on pattern analysis and machine intelligence* (2018) (cit. on p. 68).
- [Xie+16] Saining Xie, Ross B. Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. „Aggregated Residual Transformations for Deep Neural Networks“. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 5987–5995 (cit. on p. 1).
- [Xu+15] Kelvin Xu, Jimmy Ba, Ryan Kiros, et al. „Show, Attend and Tell: Neural Image Caption Generation with Visual Attention“. In: *ICML*. 2015 (cit. on p. 1).
- [Yan+18] Jianwei Yang, Jiasen Lu, Stefan Lee, Dhruv Batra, and Devi Parikh. „Visual Curiosity: Learning to Ask Questions to Learn Visual Recognition“. In: *CoRL*. 2018 (cit. on p. 78).
- [YL17] Denis Yarats and Mike Lewis. „Hierarchical text generation and planning for strategic dialogue“. In: *arXiv preprint arXiv:1712.05846* (2017) (cit. on p. 78).
- [Yos+14] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. „How transferable are features in deep neural networks?“ In: *Advances in Neural Information Processing Systems*. 2014, pp. 3320–3328 (cit. on p. 18).
- [ZE18] Tiancheng Zhao and Maxine Eskénazi. „Zero-Shot Dialog Generation with Cross-Domain Latent Actions“. In: *SIGDIAL Conference*. 2018 (cit. on p. 78).
- [Zha+16] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. „Understanding deep learning requires rethinking generalization“. In: *ICLR* (2016) (cit. on p. 3).

- [Zho+14] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. „Learning Deep Features for Scene Recognition using Places Database“. In: *NIPS*. 2014 (cit. on p. 17).
- [ZLE18] Tiancheng Zhao, Kyusong Lee, and Maxine Eskénazi. „Unsupervised Discrete Sentence Representation Learning for Interpretable Neural Dialog Generation“. In: *ACL*. 2018 (cit. on p. 78).
- [ZXE19] Tiancheng Zhao, Kaige Xie, and Maxine Eskénazi. „Rethinking Action Spaces for Reinforcement Learning in End-to-end Dialog Agents with Latent Variable Models“. In: *NAACL-HLT*. 2019 (cit. on pp. 66, 67, 76, 78).